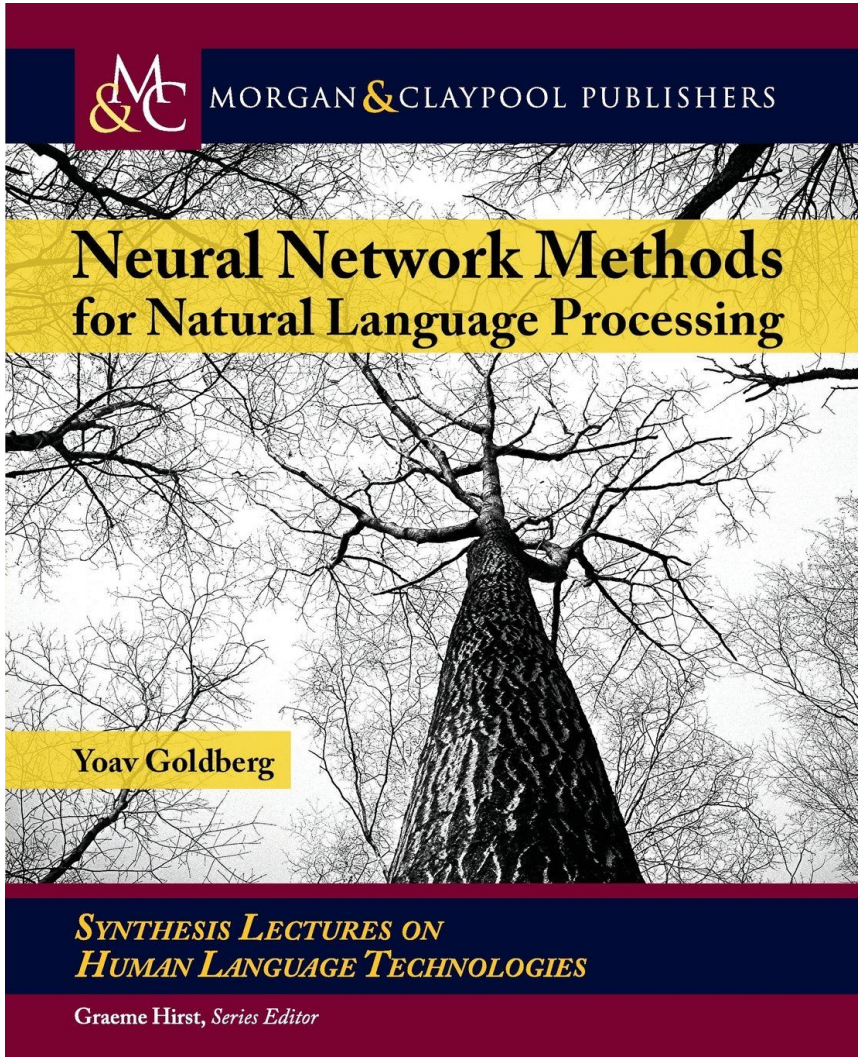
Before we start

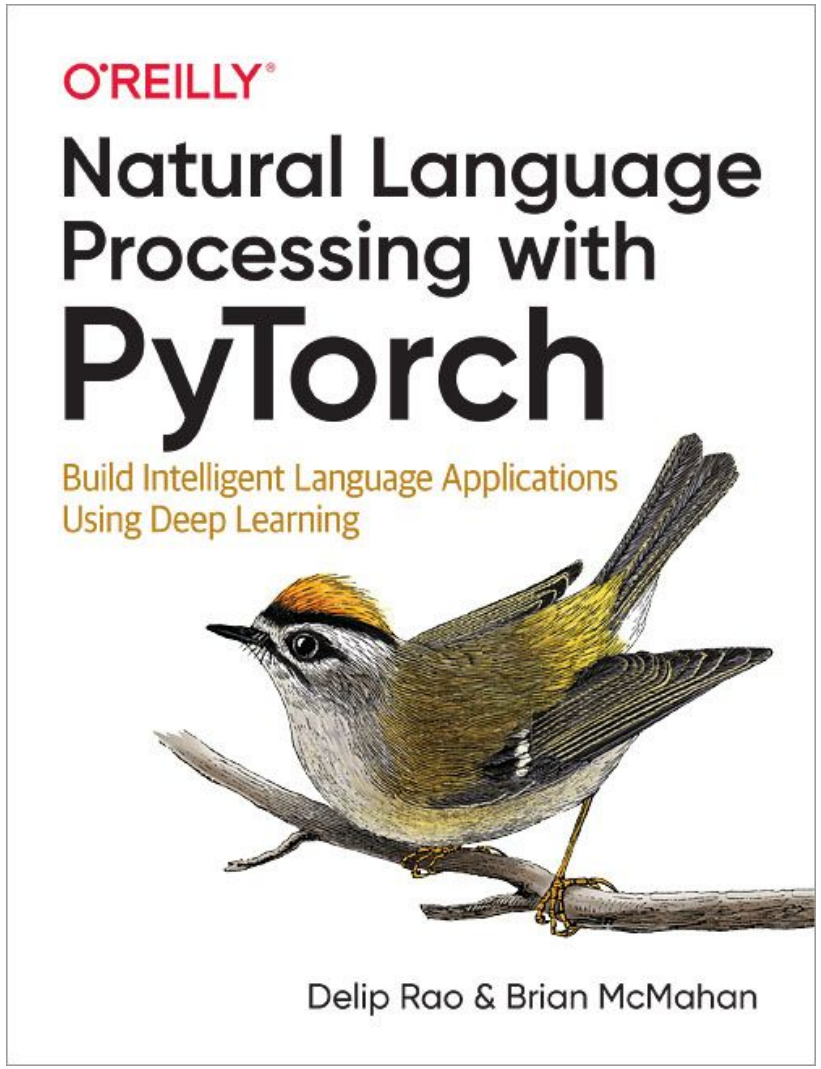
- Alfredo misses you and says “hi”!

Reference textbook 1



- Title: Neural Network Methods for Natural Language Processing
- Author: Yoav Goldberg
- Year: 2017
- 311 Pages
- Alternatively,
  - A Primer on Neural Network Models for Natural Language Processing
  - Journal of Artificial Intelligence Research
  - Freely available online

Reference textbook 2



- Title: Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning
- Author: Delip Rao
- Year: 2019

Machine Translation

Instructor: Kyunghyun Cho (NYU, Facebook)

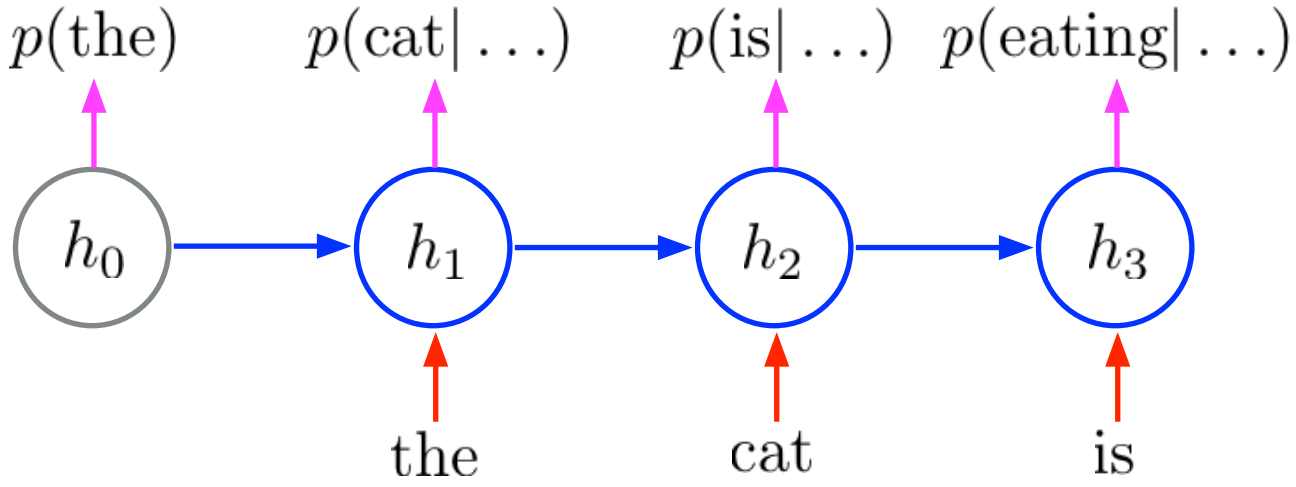
Assistants: Roberta Raileau (NYU), Ilia Kulikov (NYU), Sreyas Mohan (NYU)

Recurrent Language Modeling

Let's delve a bit deeper into a recurrent network and language modeling with it.

Recurrent Language Model

Example) $p(\text{the}, \text{cat}, \text{is}, \text{eating})$

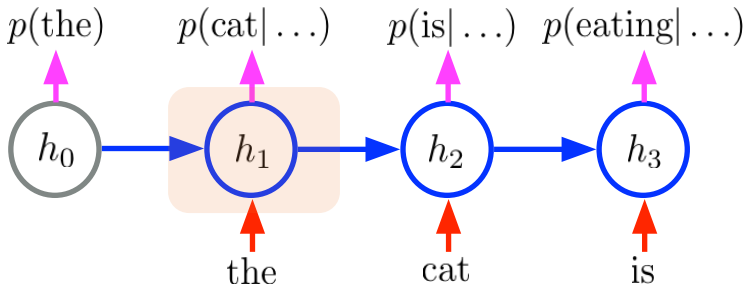


Read, *Update* *and* *Predict*

Building a Recurrent Language Model

Transition Function $h_t = f(h_{t-1}, x_{t-1})$

- Inputs
  - i. Previous word $x_{t-1} \in \{1, 2, \ldots, |V|\}$
  - ii. Previous state $h_{t-1} \in \mathbb{R}^d$
- Parameters
  - i. Input weight matrix $W \in \mathbb{R}^{|V| \times d}$
  - ii. Transition weight matrix $U \in \mathbb{R}^{d \times d}$
  - iii. Bias vector $b \in \mathbb{R}^d$
- Naïve Transition Function



$$f(h_{t-1}, x_{t-1}) = \tanh(W\,[x_{t-1}] + U h_{t-1} + b)$$

Building a Recurrent Language Model

Transition Function $h_t = f(h_{t-1}, x_{t-1})$

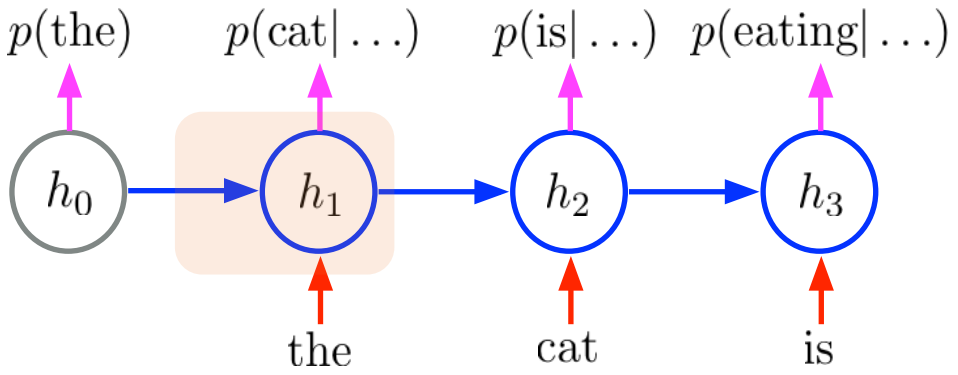
- Naïve Transition Function

$$f(h_{t-1}, x_{t-1}) = \tanh(W[x_{t-1}] + Uh_{t-1} + b)$$

Element-wise nonlinear transformation

Linear transformation of previous state

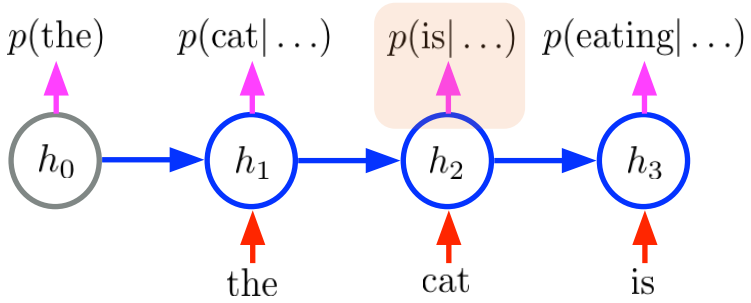
Trainable word vector



Building a Recurrent Language Model

Readout Function $p(x_t = w|x_{<t}) = g_w(h_t)$

- Inputs
  i. Current state $h_t \in \mathbb{R}^d$
- Parameters
  i. Readout weight matrix $R \in \mathbb{R}^{|V| \times d}$
  ii. Bias vector $c \in \mathbb{R}^{|V|}$



- Softmax Readout

$$p(x_t = w|x_{<t}) = g_w(h_t) = \frac{\exp(R\,[w]^\top h_t + c_w)}{\sum_{i=1}^{|V|} \exp(R\,[i]^\top h_t + c_i)}$$

Building a Recurrent Language Model

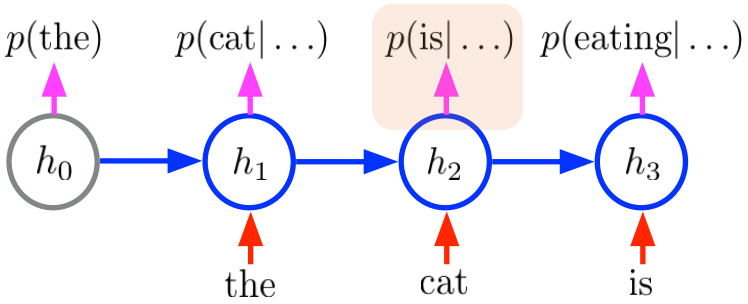
Readout Function $p(x_t = w|x_{<t}) = g_w(h_t)$

$$p(x_t = w|x_{<t}) = g_w(h_t) = \frac{\exp(R[w]^\top h_t + c_w)}{\sum_{i=1}^{|V|} \exp(R[i]^\top h_t + c_i)}$$

Exponentiation

Normalization

Compatibility between a trainable word vector and the hidden state



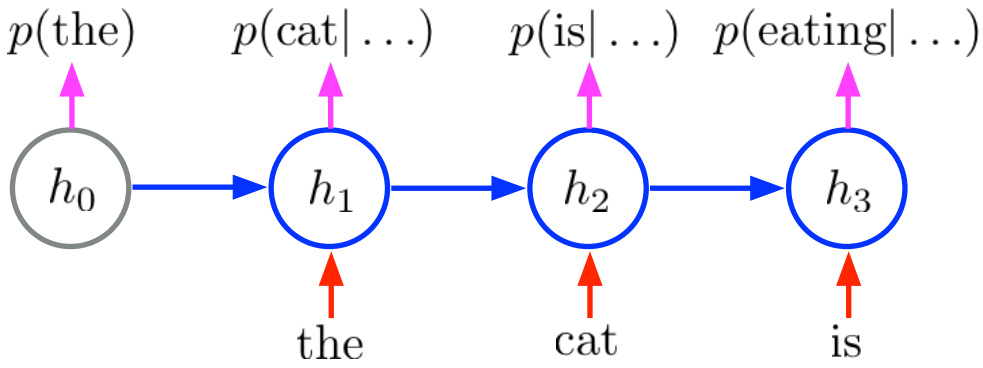
Training a Recurrent Language Model

- Log-probability of one training sentence

$$\log p(x_1^n, x_2^n, \ldots, x_{T^n}^n) = \sum_{t=1}^{T^n} \log p(x_t^n | x_1^n, \ldots, x_{t-1}^n)$$

- Training set $D = \{X^1, X^2, \ldots, X^N\}$
- Log-likelihood Functional

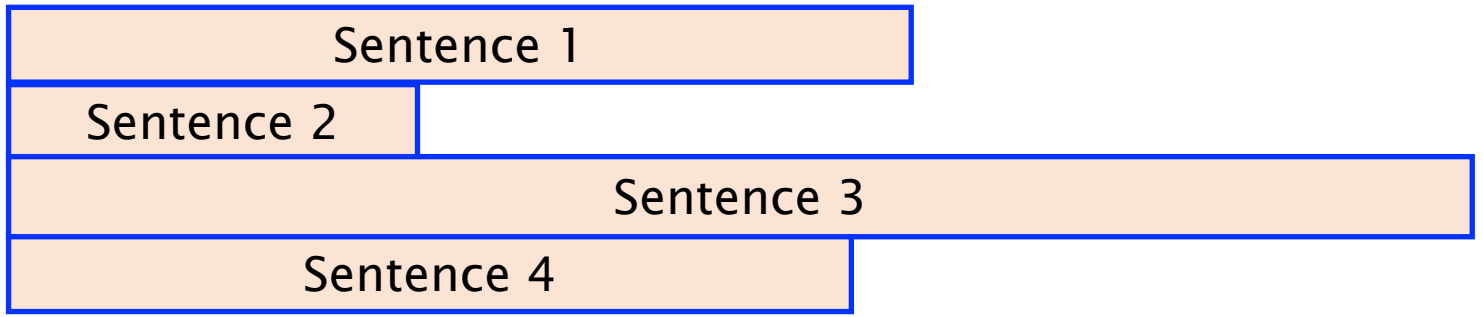
$$\mathcal{L}(\theta, D) = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T^n} \log p(x_t^n | x_1^n, \ldots, x_{t-1}^n)$$



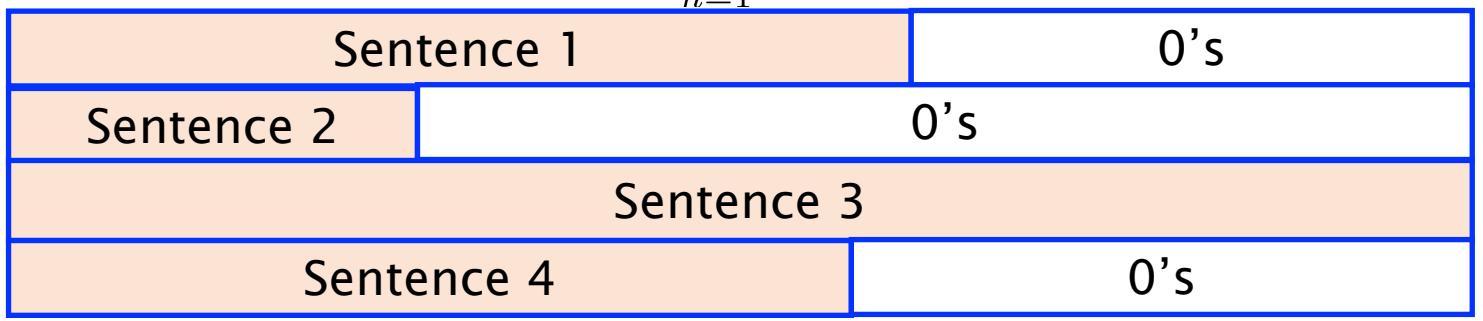
Minimize $-\mathcal{L}(\theta, D)$!!

Minibatch Stochastic Gradient Descent

- Building a minibatch is not trivial due to length differences.

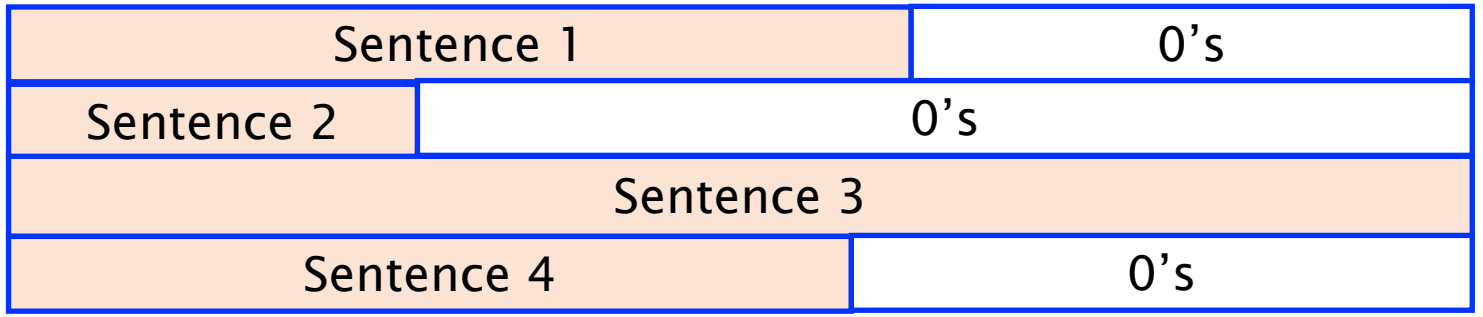


1. Padding and Masking: *suitable for GPU's, but wasteful*
   - *Wasted computation:* $\sum_{n=1}^{N} \max_{n'=1,\ldots,N} l(X^{n'}) - l(X^{n})$

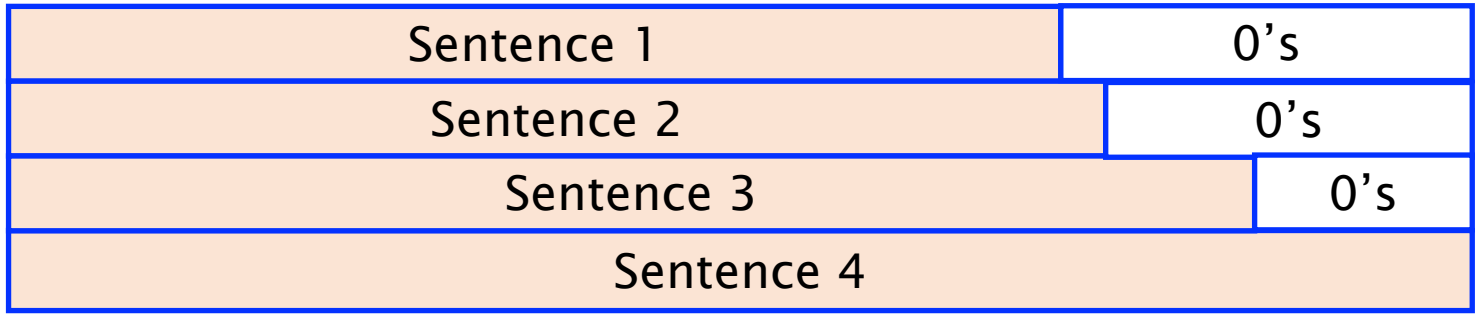


Minibatch Stochastic Gradient Descent

1. Padding and Masking: *suitable for GPU's, but wasteful*
   - *Wasted computation:* $\sum_{n=1}^{N} \max_{n'=1,\ldots,N} l(X^{n'}) - l(X^{n})$

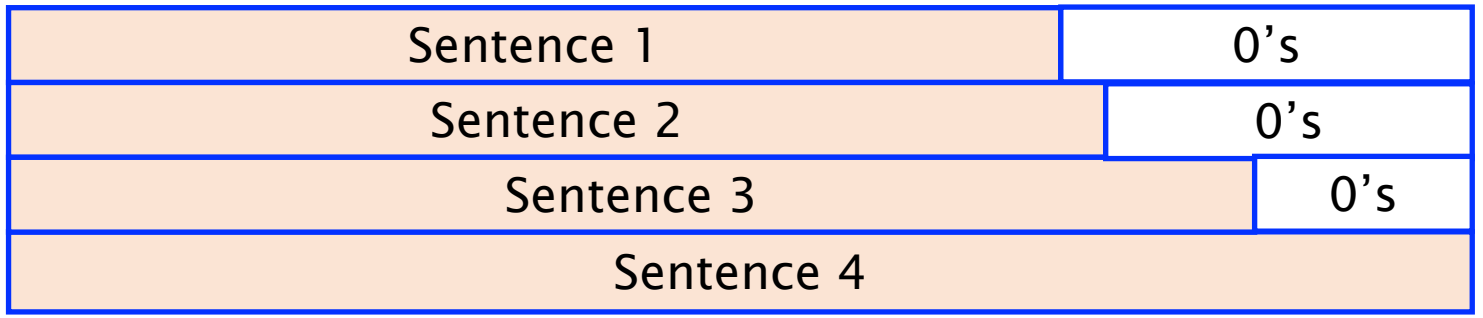


2. Smarter Padding and Masking: *minimize the waste*
   - *Ensure that the length differences are minimal.*
   - *Sort the sentences and sequentially build a minibatch*

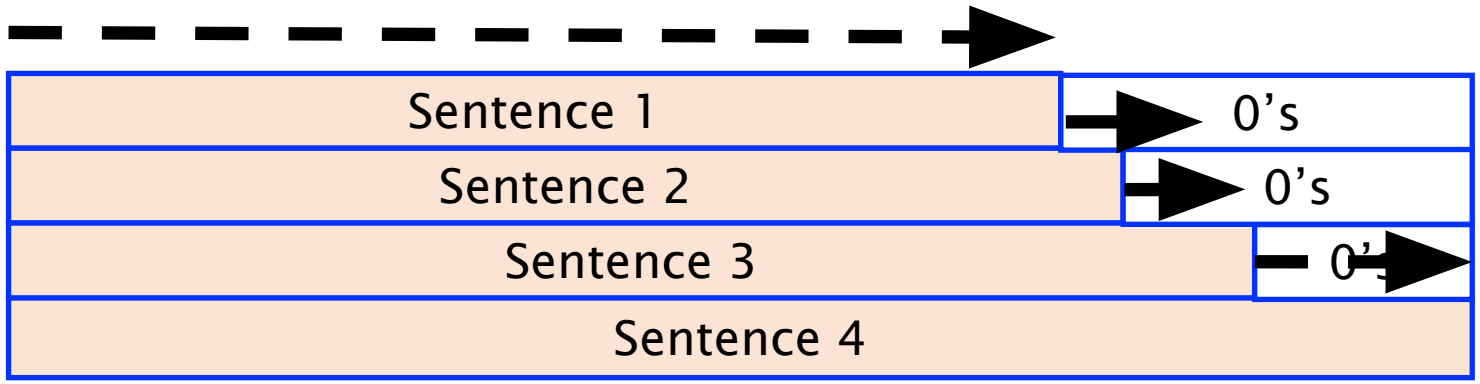


Minibatch Stochastic Gradient Descent

2. Smarter Padding and Masking: *minimize the waste*



3. Smarter Padding and Smarter Stopping: *toward zero waste*
   - *Drop a finished sentence from a minibatch*
   - *Not as flexible…*

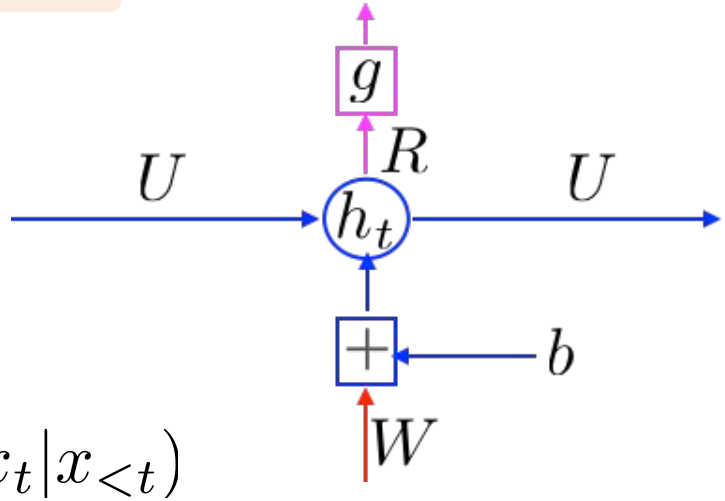


Backpropagation through Time

How do we compute $\nabla \mathcal{L}(\theta, X)$ *?*

- Decompose the per-sample cost into per-step cost functions

$$\nabla \mathcal{L}(\theta, X) = \sum_{t=1}^{T} \nabla \log p(x_t | x_{<t}, \theta)$$



- Compute per-step cost function from time $t = T$
  1. Cost derivative $\partial \log p(x_t | x_{<t}) / \partial g$
  2. Gradient w.r.t. R : $\times \partial g / \partial R$
  3. Gradient w.r.t. h_t : $\times \partial g / \partial h_t + \partial h_{t+1} / \partial h_t \log p(x_t | x_{<t})$
  4. Gradient w.r.t. U : $\times \partial h_t / \partial U$
  5. Gradient w.r.t. b and W : $\times \partial h_t / \partial b$ and $\times \partial h_t / \partial W$
  6. Accumulate the gradient and $t \leftarrow t - 1$

Note: *I'm abusing math a lot here!!*

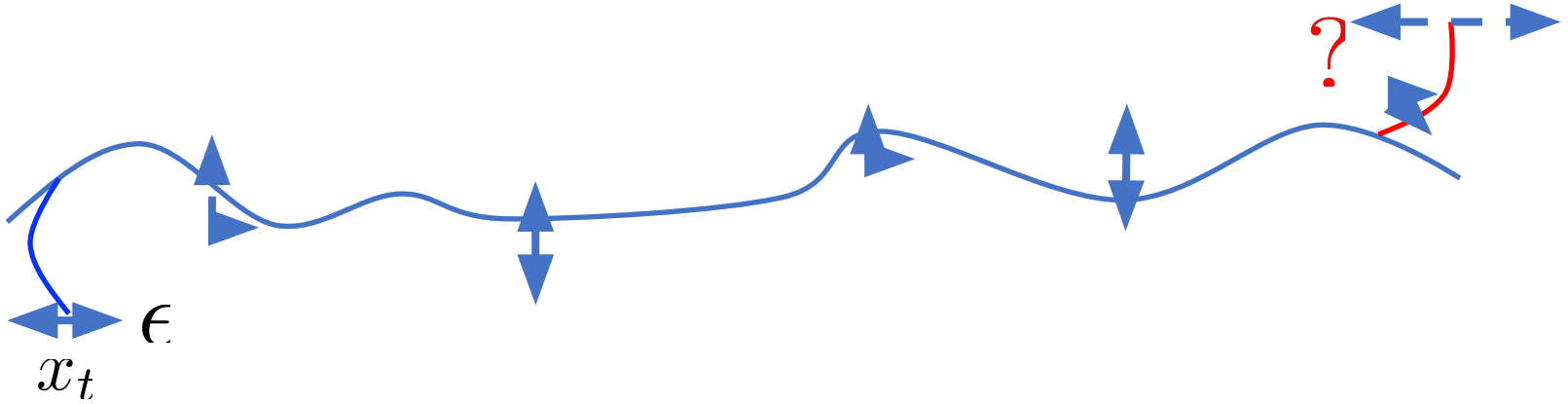
Backpropagation through Time

Intuitively, what's happening here?

1. Measure the influence of the past on the future

$$\frac{\partial \log p(x_{t+n}|x_{<t+n})}{\partial h_t} = \frac{\partial \log p(x_{t+n}|x_{<t+n})}{\partial g} \frac{\partial g}{\partial h_{t+n}} \frac{\partial h_{t+n}}{\partial h_{t+n-1}} \cdots \frac{\partial h_{t+1}}{\partial h_t} \frac{\partial h_t}{\partial x_t}$$

2. If I perturb the input at t, how does it affect $p(x_{t+n}|x_{<t+n})$?



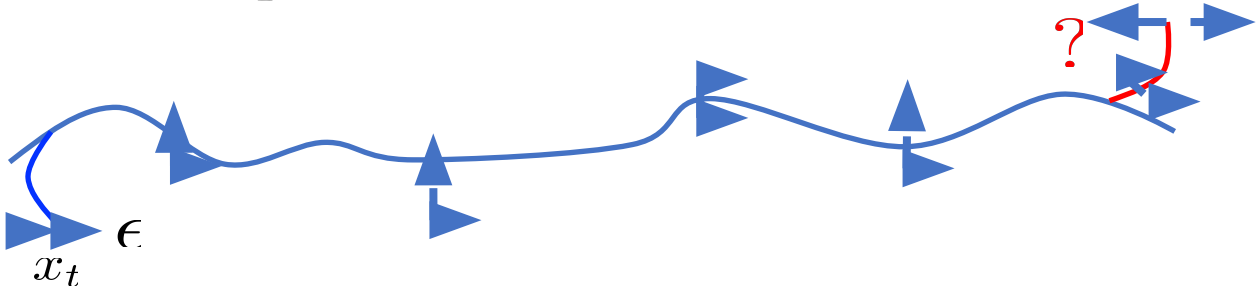
Backpropagation through Time

Intuitively, what's happening here?

1. Measure the influence of the past on the future

$$\frac{\partial \log p(x_{t+n}|x_{<t+n})}{\partial h_t} = \frac{\partial \log p(x_{t+n}|x_{<t+n})}{\partial g} \frac{\partial g}{\partial h_{t+n}} \frac{\partial h_{t+n}}{\partial h_{t+n-1}} \cdots \frac{\partial h_{t+1}}{\partial h_t} \frac{\partial h_t}{\partial x_t}$$

2. If I perturb the input at t, how does it affect $p(x_{t+n}|x_{<t+n})$?



3. Change the parameters θ so as to maximize $p(x_{t+n}|x_{<t+n})$

Backpropagation through Time

Intuitively, what's happening here?

1. Measure the influence of the past on the future

$$\frac{\partial \log p(x_{t+n}|x_{<t+n})}{\partial h_t} = \frac{\partial \log p(x_{t+n}|x_{<t+n})}{\partial g} \frac{\partial g}{\partial h_{t+n}} \frac{\partial h_{t+n}}{\partial h_{t+n-1}} \cdots \frac{\partial h_{t+1}}{\partial h_t}$$

2. With a naïve transition function

$$f(h_{t-1}, x_{t-1}) = \tanh(W\,[x_{t-1}] + U h_{t-1} + b)$$

We get $$\frac{\partial J_{t+n}}{\partial h_t} = \frac{\partial J_{t+n}}{\partial g} \frac{\partial g}{\partial h_{t+N}} \underbrace{\prod_{n=1}^{N} U^\top \operatorname{diag}\left(\frac{\partial \tanh(a_{t+n})}{\partial a_{t+n}}\right)}$$

Problematic!

– Bengio et al. (1994)

Backpropagation through Time

*Gradient either **vanishes** or **explodes*** Pascanu et al. (2013)

- What happens? $$\frac{\partial J_{t+n}}{\partial h_t} = \frac{\partial J_{t+n}}{\partial g}\frac{\partial g}{\partial h_{t+N}} \underbrace{\prod_{n=1}^{N} U^\top \mathrm{diag}\left(\frac{\partial \tanh(a_{t+n})}{\partial a_{t+n}}\right)}$$

1. The gradient *likely* explodes if

$$e_{\max} \geq \frac{1}{\max \tanh'(x)} = 1$$

2. The gradient *likely* vanishes if

$$e_{\max} < \frac{1}{\max \tanh'(x)} = 1$$

$e_{\max}$: largest eigenvalue of U

Backpropagation through Time

Let the (norm of the) gradient explode!

- *"when gradients explode so does the curvature along v, leading to a wall in the error surface"*

- Simple solution: Gradient Clipping
  1. Norm clipping

$$\tilde{\nabla} \leftarrow \begin{cases} \frac{c}{\|\nabla\|}\nabla & \text{,if } \|\nabla\| \geq c \\ \nabla & \text{,otherwise} \end{cases}$$

  2. Element-wise clipping

$$\nabla_i \leftarrow \min(c, \nabla_i), \text{ for all } i \in \{1, \ldots, \dim\nabla\}$$

Pascanu et al. (2013)

Backpropagation through Time

Vanishing gradient is super-problematic

- We cannot tell whether
  1. no long-term dependency between *t* and *t*+*n* in data, or
  2. wrong configuration of parameters:

$$e_{\max}(U) < \frac{1}{\max \tanh'(x)}$$

- We only observe $\left\| \frac{\partial h_{t+N}}{\partial h_t} \right\| = \left\| \prod_{n=1}^{N} U^\top \operatorname{diag}\left(\frac{\partial \tanh(a_{t+n})}{\partial a_{t+n}} \right) \right\| \to 0$

Backpropagation through Time

Vanishing gradient is super-problematic

- Let's just say there is such a long-term dependency. Then,
  - "*we … force the network to increase the norm of* $\frac{\partial h_{t+N}}{\partial h_t}$ *at the expense of larger errors*"

Pascanu et al. (2013)

- This can be done by regularizing

$$\sum_{t=1}^{T} \left(1 - \frac{\left\| \frac{\partial \tilde{C}}{\partial \mathbf{h}_{t+1}} \frac{\partial \mathbf{h}_{t+1}}{\partial \mathbf{h}_t} \right\|}{\left\| \frac{\partial \tilde{C}}{\partial \mathbf{h}_{t+1}} \right\|} \right)^2$$

- This doesn't seem like a great nor easy way to deal with the vanishing gradient.

Gated Recurrent Unit

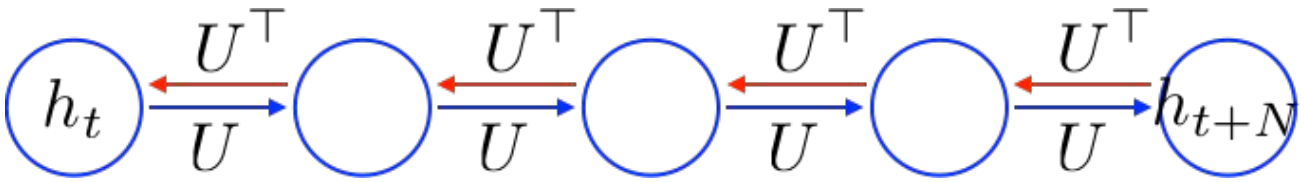
- Perhaps, the problem is with the naïve transition function

$$f(h_{t-1}, x_{t-1}) = \tanh(W\,[x_{t-1}] + U h_{t-1} + b)$$

- With it, the temporal derivative is

$$\frac{\partial h_{t+1}}{\partial h_t} = U^\top \frac{\partial \tanh(a)}{\partial a}$$

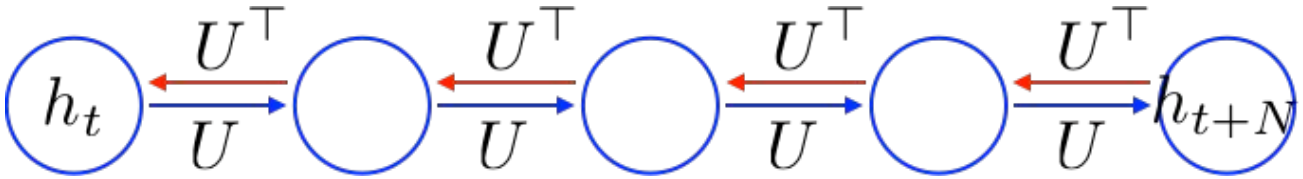
- It implies that the error must backpropagate through all the intermediate nodes:



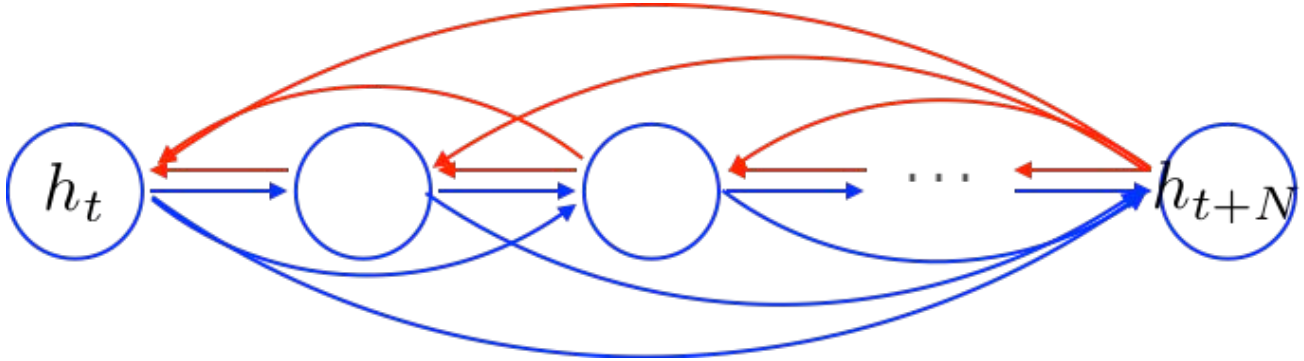
Gated Recurrent Unit

$$f(h_{t-1}, x_{t-1}) = \tanh(W\,[x_{t-1}] + Uh_{t-1} + b)$$

- It implies that the error must backpropagate through all the intermediate nodes:



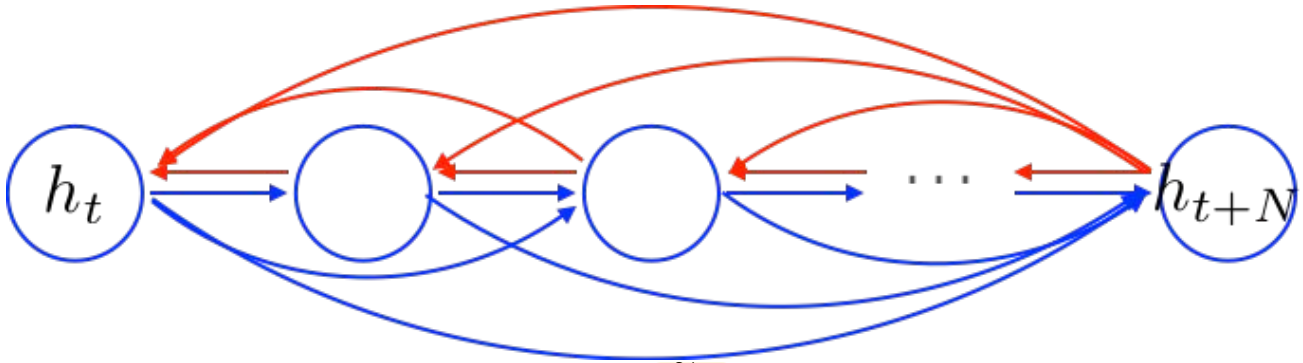
- Perhaps we can create shortcut connections.



Gated Recurrent Unit

$$f(h_{t-1}, x_{t-1}) = \tanh(W\,[x_{t-1}] + U h_{t-1} + b)$$

- Perhaps we can create *adaptive* shortcut connections.



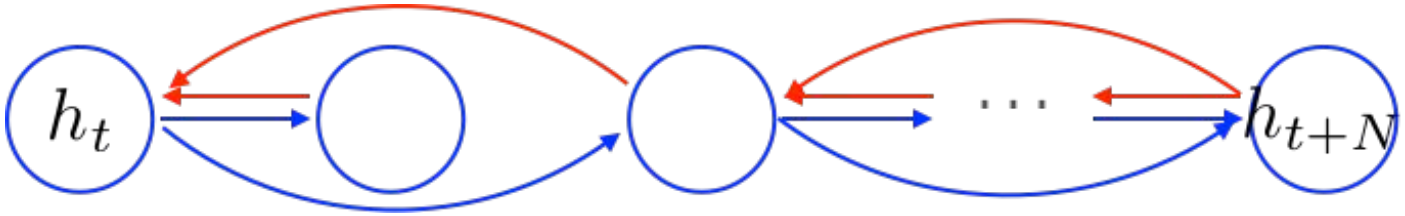
$$f(h_{t-1}, x_{t-1}) = u_t \odot \tilde{h}_t + (1 - u_t) \odot h_{t-1}$$

- Candidate Update $\tilde{h}_t = \tanh(W\,[x_{t-1}] + U h_{t-1} + b)$
- Update gate $u_t = \sigma(W_u\,[x_{t-1}] + U_u h_{t-1} + b_u)$

Gated Recurrent Unit

$$f(h_{t-1}, x_{t-1}) = \tanh(W\,[x_{t-1}] + U h_{t-1} + b)$$

- We also let the network prune unnecessary shortcuts *adaptively*.

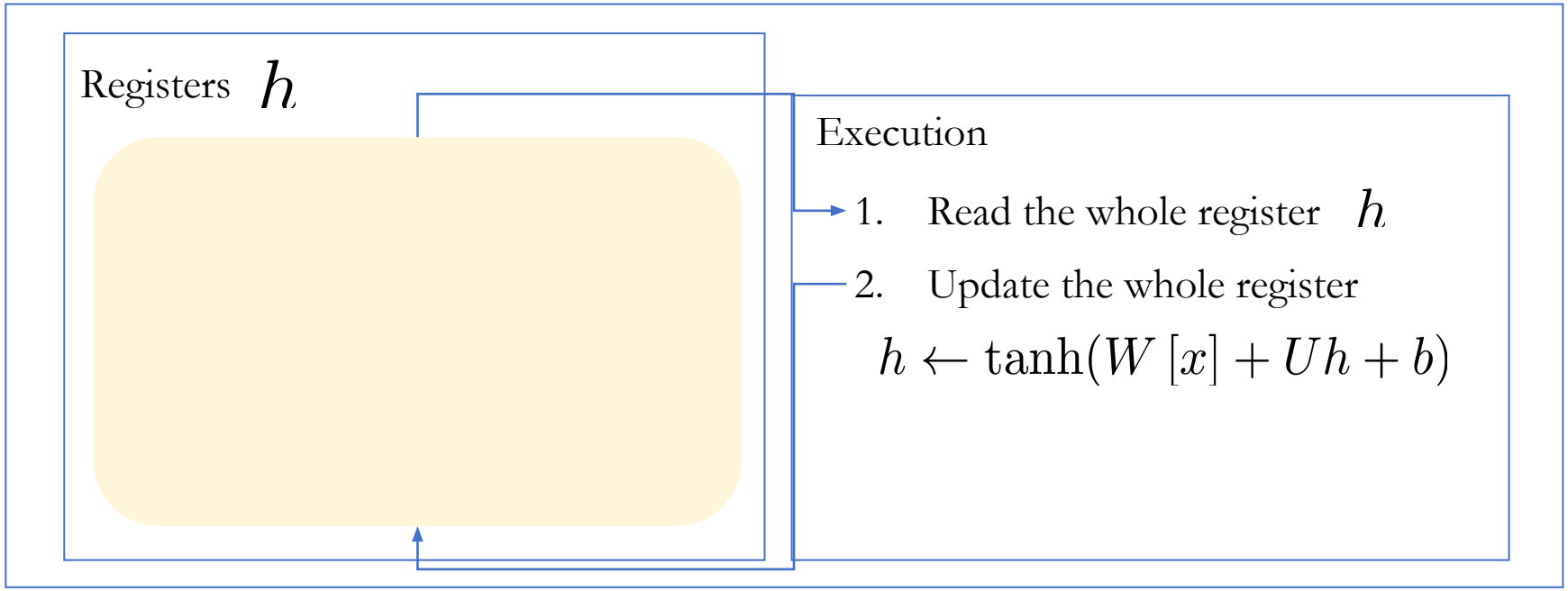


$$f(h_{t-1}, x_{t-1}) = u_t \odot \tilde{h}_t + (1 - u_t) \odot h_{t-1}$$

- Candidate Update $\tilde{h}_t = \tanh(W x_{t-1} + U(r_t \odot h_{t-1}) + b)$
- Reset gate $r_t = \sigma(W_r x_{t-1} + U_r h_{t-1} + b_r)$
- Update gate $u_t = \sigma(W_u\,[x_{t-1}] + U_u h_{t-1} + b_u)$

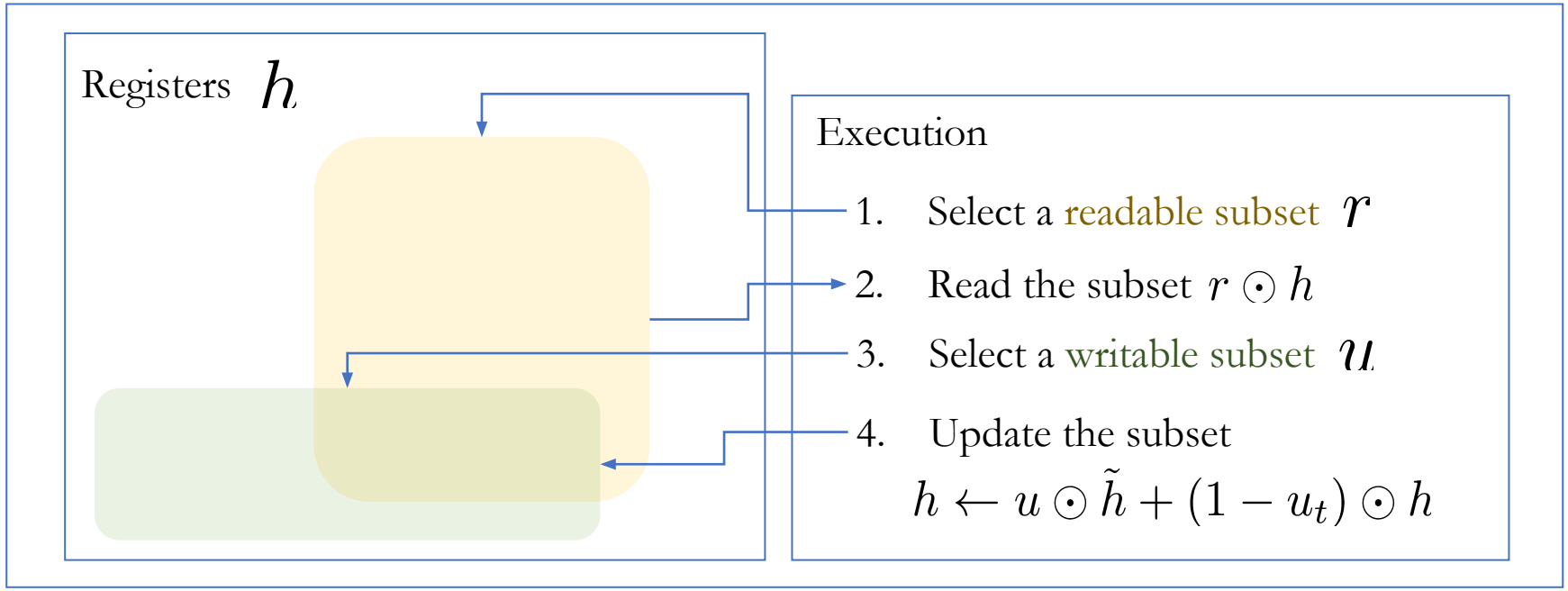
Gated Recurrent Unit

tanh-RNN vs CPU



Gated Recurrent Unit

GRU vs CPU



Clearly gated recurrent units* are much more realistic.

** By gated recurrent units, I refer to both LSTM and GRU.*

Machine Translation: a Natural Extension of Neural Language Modeling

You have already learned how to build it.

Machine Translation

- Input: a sentence written in a source language L_S
- Output: a corresponding sentence in a target language L_T
- Problem statement:
  - Supervised learning: given the input sentence, output its translation
  - Compute the conditional distribution over all possible translation given the input

    $p(Y = (y_1, \ldots, y_T) | X = (x_1, \ldots, x_{T'}))$

- *We have already learned every necessary ingredient for building a full neural machine translation system.*

Token Representation – One-hot Vectors

1. Build source and target vocabularies of unique tokens
   - For each of source and target languages,
     1. Tokenize: separate punctuations, normalize punctuations, …
        e.g., “I’m going” => (“I”, “’m”, “going”), replace ‘, ’, `, into “`”, …
        use Spacy.io, NLTK or Moses’ tokenizer.
     2. Subword segmentation: segment each token into a sequence of subwords
        e.g., “going” => (“go”, “ing”), use BPE [Sennrich et al., 2015]
     3. Collect all unique subwords, sort them by their frequencies (descending) and assign indices.
2. Transform each subword token into a corresponding one-hot vector.*

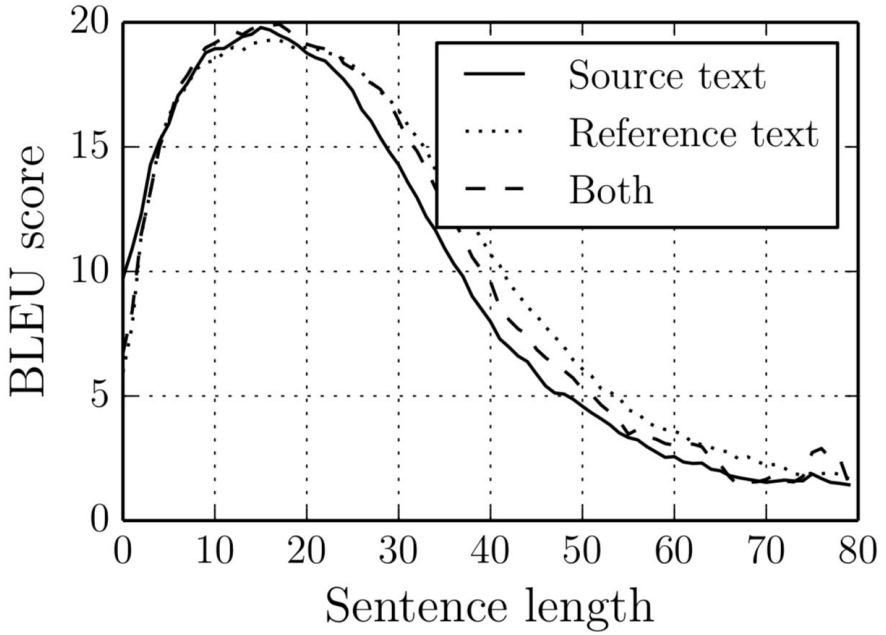
* Of course, don’t do it offline. Transform them online.

Encoder – Source Sentence Representation

- Encode the source sentence into a set of sentence representation vectors
  - # of encoded vectors is proportional to the source sentence length: often same. $H = (h_1, \ldots, h_{T'})$
  - Recurrent networks have been widely used [Cho et al., 2014; Sutskever et al., 2014], but CNN [Gehring et al., 2017; Kalchbrenner&Blunsom, 2013] and self-attention [Vaswani et al., 2017] are used increasingly more often. See Lecture 2 for details.
- We do not want to collapse them into a single vector.
  - Collapsing often corresponds to information loss.
  - Increasingly more difficult to encode the entire source sentence into a single vector, as the sentence length increases [Cho et al., 2014b].
  - We didn't know initially until [Bahdanau et al., 2015].

Encoder – Source Sentence Representation

- Encode the source sentence into a set of sentence representation vectors
- We do not want to collapse them into a single vector.
  - Increasingly more difficult to encode the entire source sentence into a single vector, as the sentence length increases [Cho et al., 2014b].



Encoder – Source Sentence Representation

- Encode the source sentence into a set of sentence representation vectors
- We do not want to collapse them into a single vector.
  - Increasingly more difficult to encode the entire source sentence into a single vector, as the sentence length increases [Cho et al., 2014b].
  - When collapsed, the system fails to translate a long sentence correctly.
    - **Source**: *An admitting privilege is the right of a doctor to admit a patient to a hospital or a medical centre to carry out a diagnosis or a procedure, based on his status as a health care worker at a hospital.*
    - **When collapsed:** *Un privilège d'admission est le droit d'un médecin de reconnaître un patient à l'hôpital ou un centre médical d'un diagnostic ou de prendre un diagnostic en fonction de son état de santé.*
  - The system translates reasonable up to a certain point, but starts drifting away.

Decoder – Language Modelling

- Autoregressive Language modelling with an infinite context $n \to \infty$
  - Larger context is necessary to generate a coherent sentence.
    - Semantics could be largely provided by the source sentence,
      but syntactic properties need to be handled by the language model directly.
  - Recurrent networks, self-attention and (dilated) convolutional networks
    - Causal structure must be followed.
    - See Lecture 3.
- **Conditional** Language modelling
  - The context based on which the next token is predicted is **two-fold**

$$p(Y|X) = \prod_{t=1}^{T} p(y_t|y_{<t}, X)$$

Decoder – Conditional Language Modelling

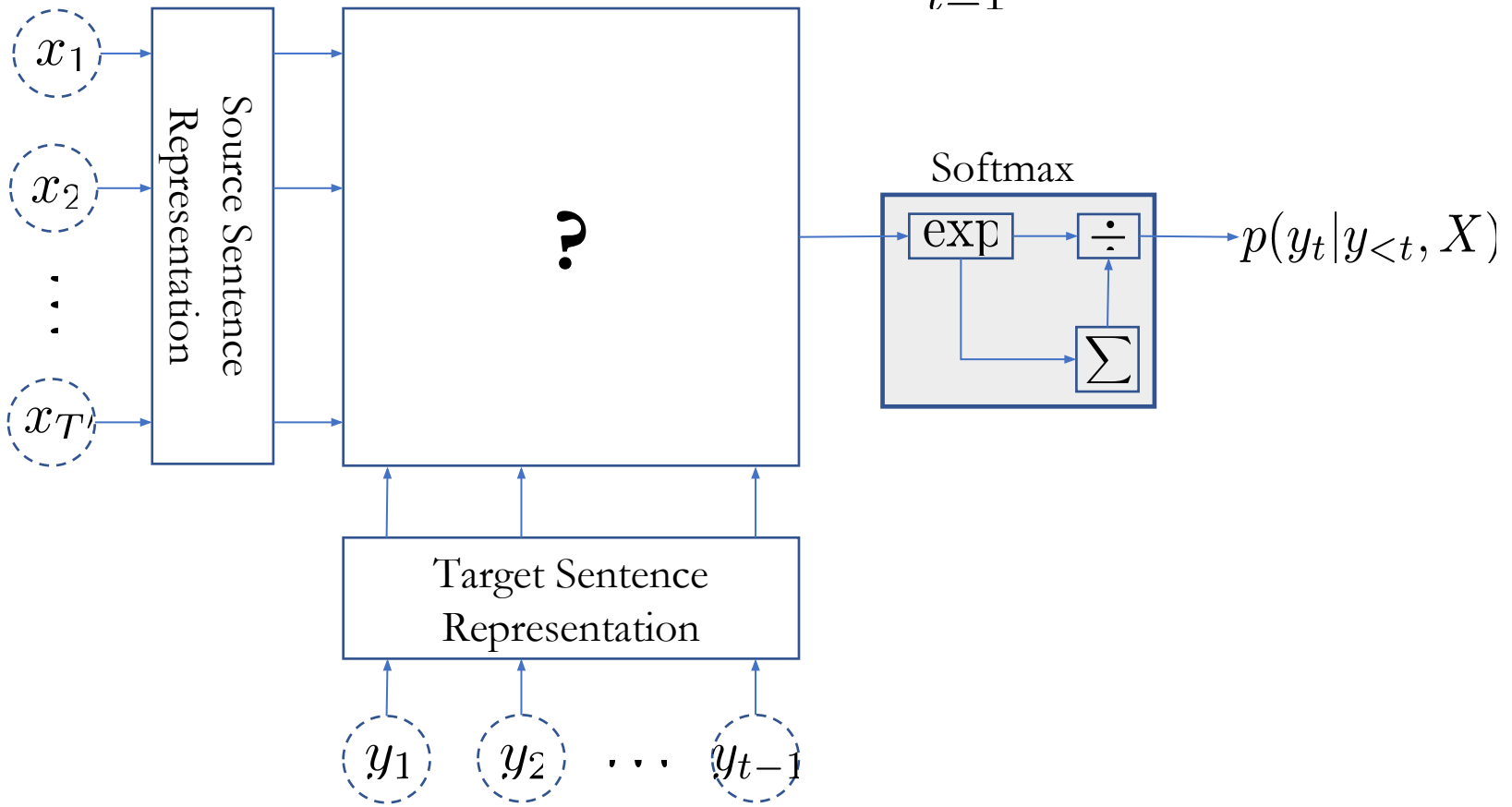
- Conditional Language modelling
  - The context based on which the next token is predicted is **two-fold**.

$$p(Y|X) = \prod_{t=1}^{T} p(y_t|y_{<t}, X)$$

- Supervised learning: T input-output training pairs per sentence
  - Input: the entire source sentence X and the preceding target tokens $y_{<t}$
  - Output: the next token y_t

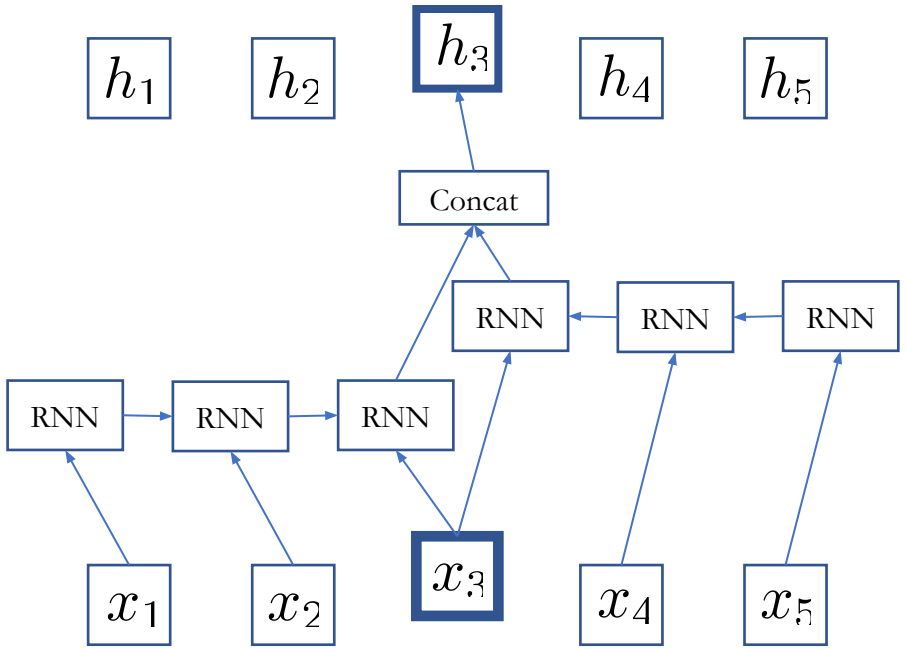
Decoder – Conditional Language Modelling

- Conditional Language modelling $p(Y|X) = \prod_{t=1}^{T} p(y_t|y_{<t}, X)$

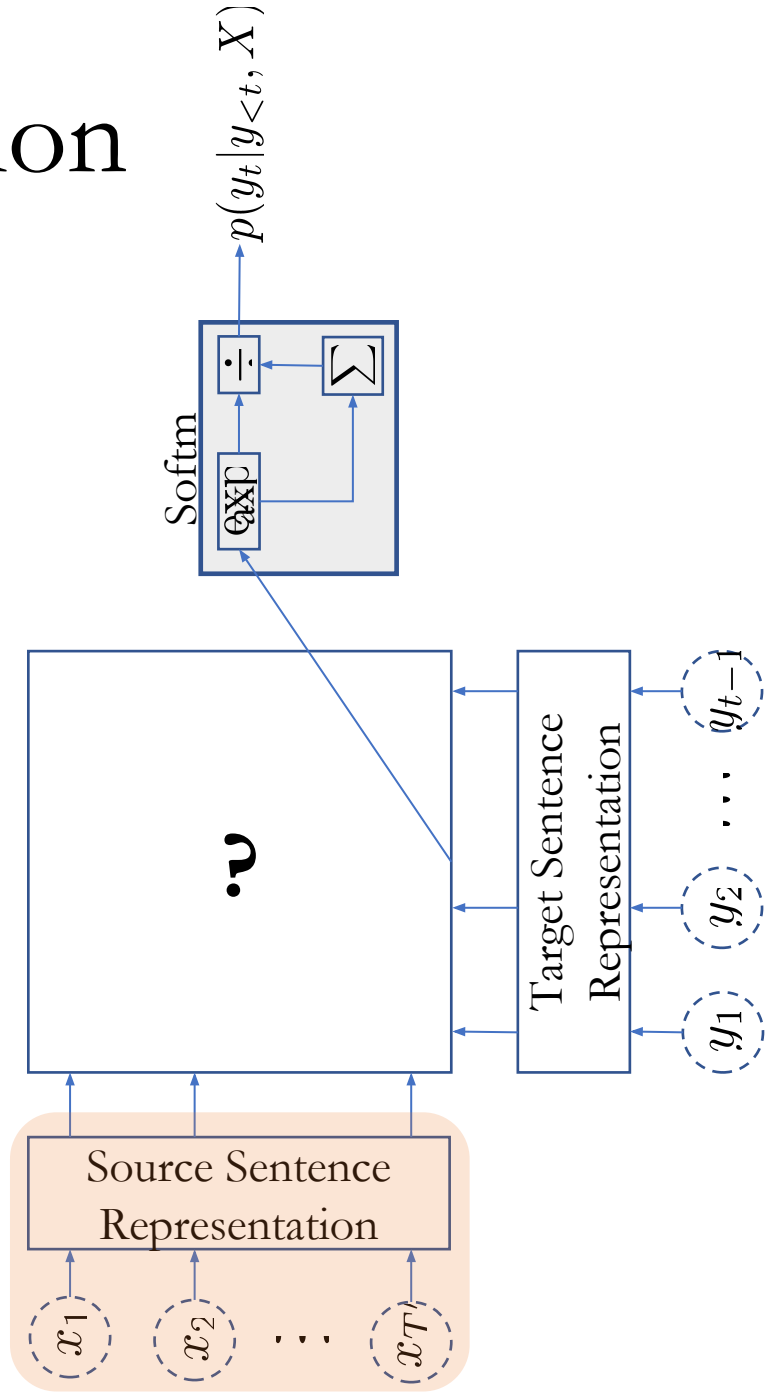


RNN Neural Machine Translation [Bahdanau et al., 2015]

1. Source sentence representation
   - A stack of bidirectional RNN's

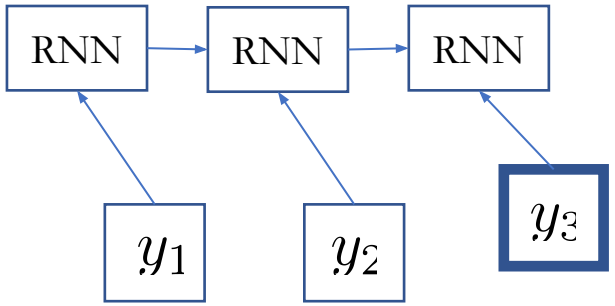


   - The extracted vector at each location is a **context-dependent vector representation**.



RNN Neural Machine Translation [Bahdanau et al., 2015]

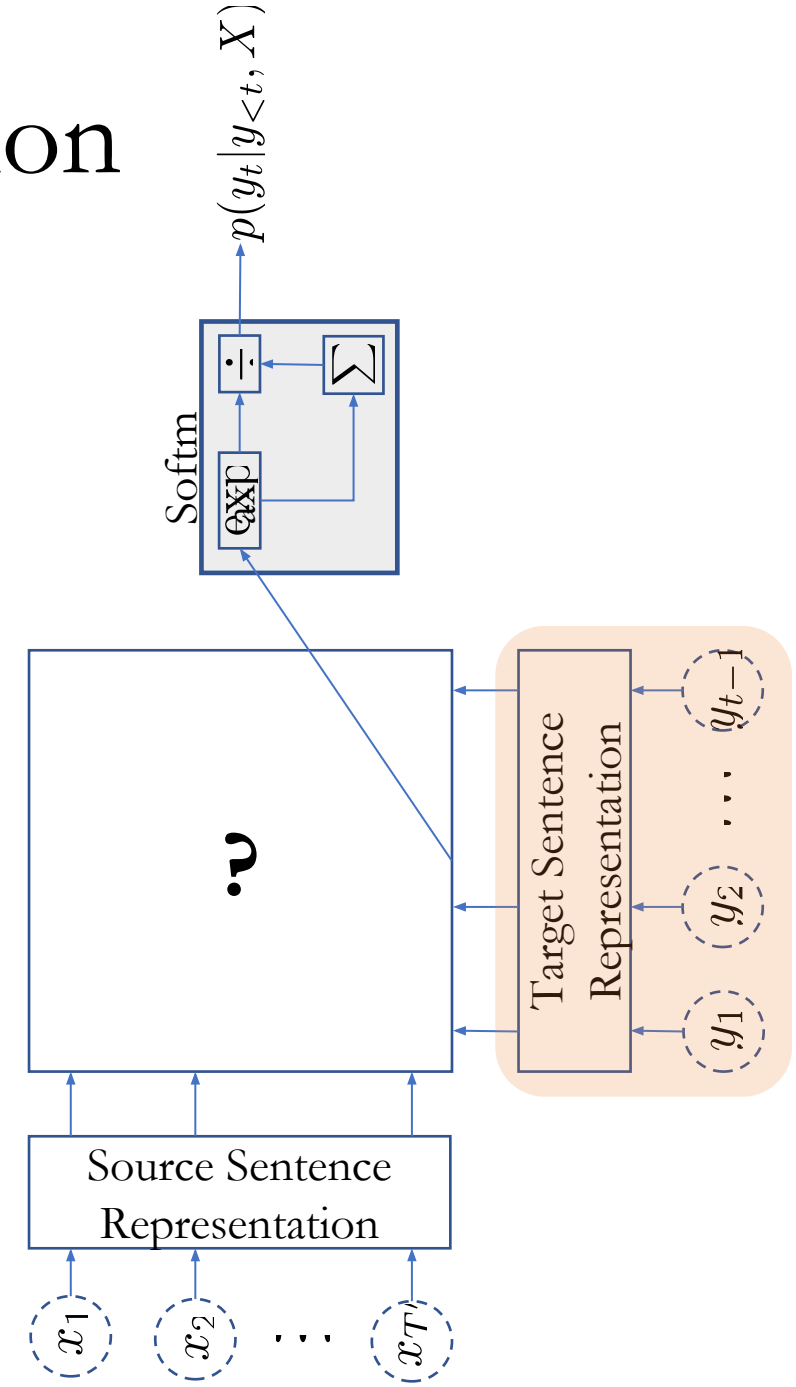
2. Target prefix representation
   - A unidirectional recurrent network



   - Compression of the target prefix

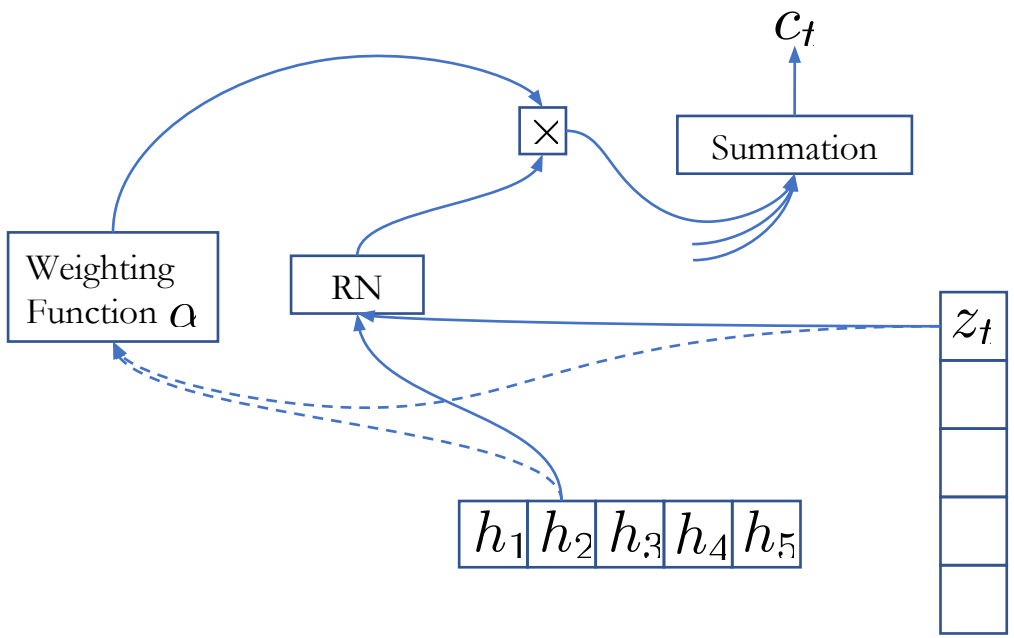
$$z_t = \mathrm{RNN}_{\mathrm{decoder}}(z_{t-1}, y_{t-1})$$

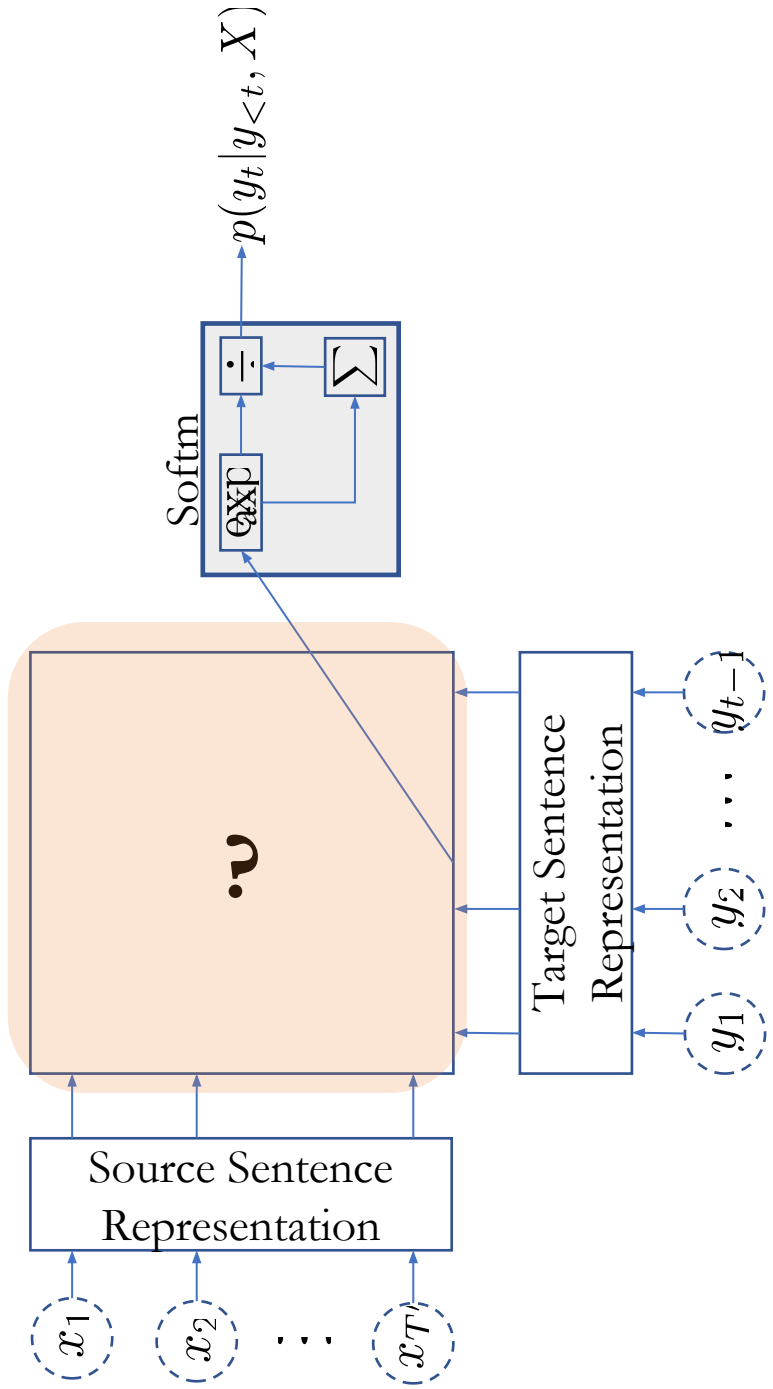
   - Summarizes what has been translated so far



RNN Neural Machine Translation [Bahdanau et al., 2015]

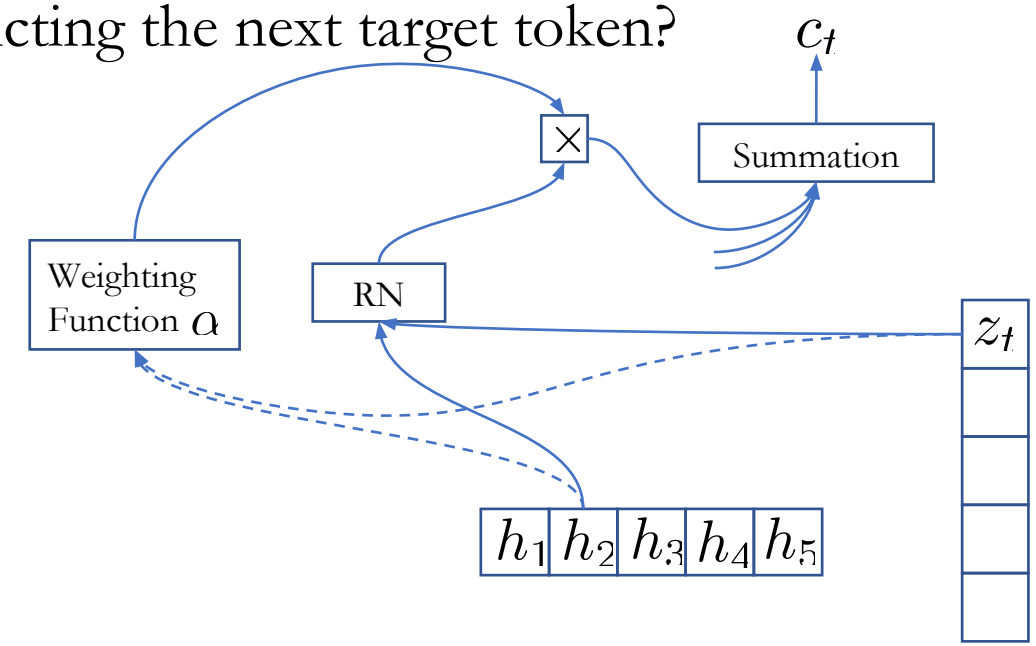
3. Attention mechanism
   - Which part of the source sentence is relevant for predicting the next target token?
   - Recall self-attention from Lecture 2



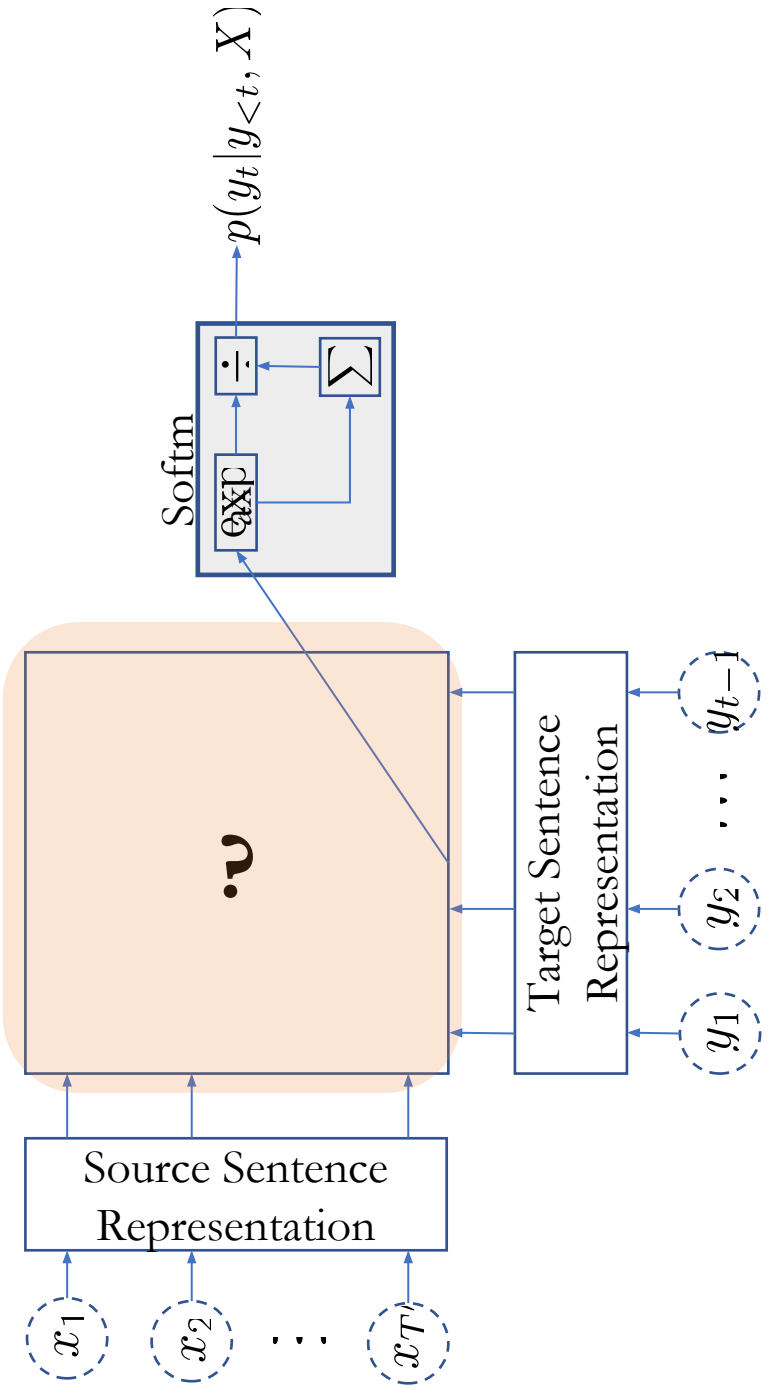


RNN Neural Machine Translation
[Bahdanau et al., 2015]

3. Attention mechanism
   - Which part of the source sentence is relevant for predicting the next target token?

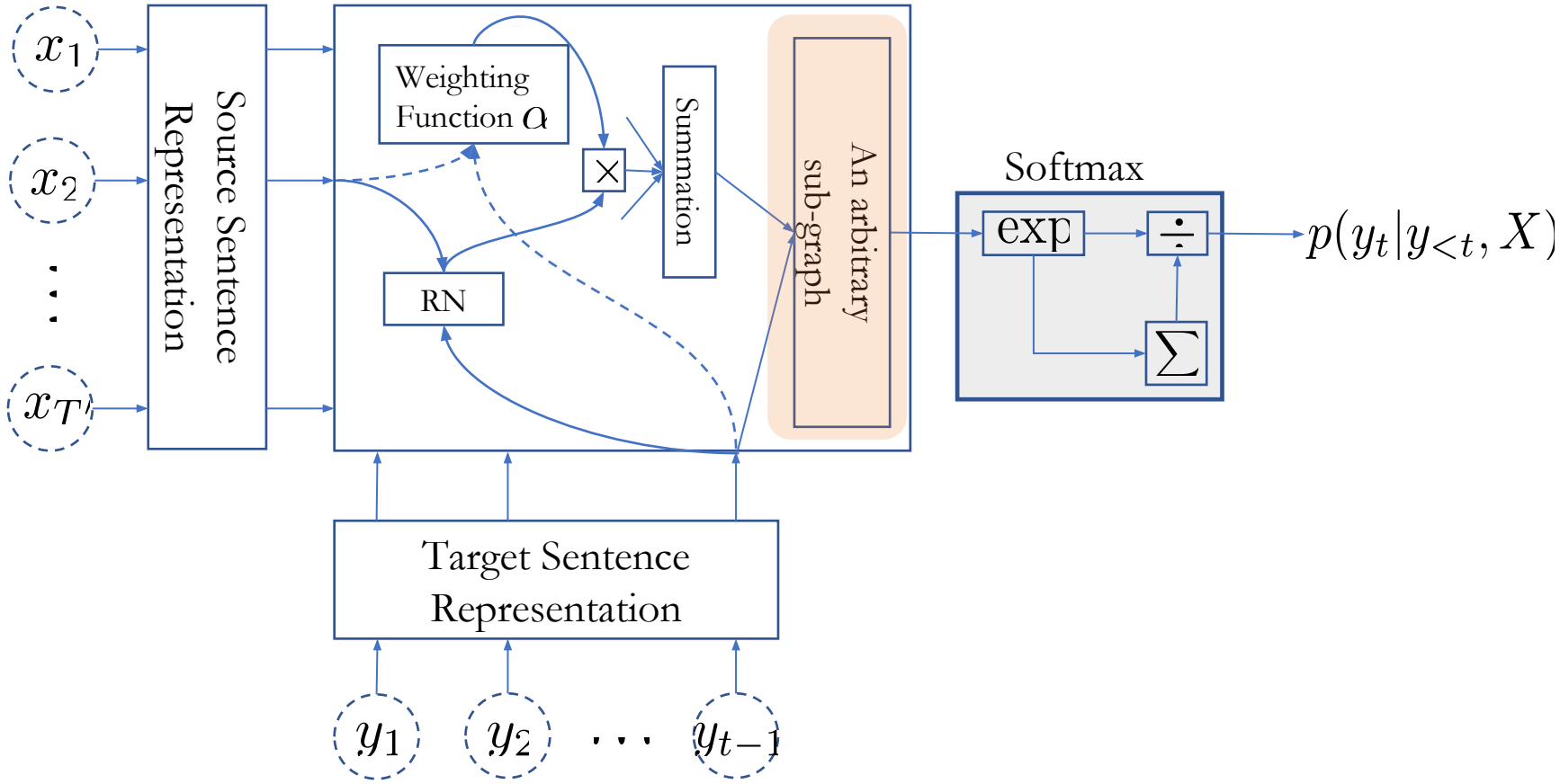


   - Time-dependent source context vector c_t



RNN Neural Machine Translation
[Bahdanau et al., 2015]

4. Fuse the source context vector and target prefix vector
   - Combines z_t and c_t into a single vector

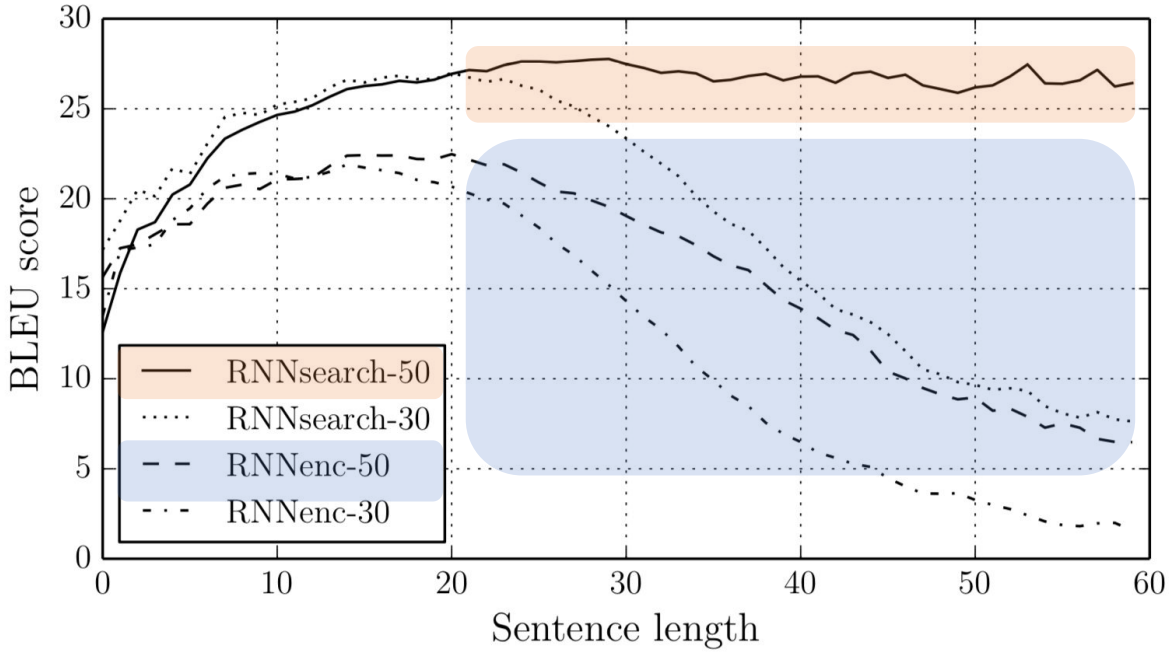


RNN Neural Machine Translation

- Conceptual process
  1. Encode: read the entire source sentence to know what to translate
  2. Attention: at each step, decide which source token(s) to translate next
  3. Decode: based on what has been translated and what need to be translated, predict the next target token.
  4. Repeat 2-3 until the <end-of-sentence> special token is generated.

RNN Neural Machine Translation

- The model is not pressured to compress the entire source sentence into a single, fixed-size vector:
  - Greatly improves the translation quality, especially of long sentences.
  - Much more efficient: less parameters are necessary.
- Bahdanau et al. [2015] showed for the first time the machine translation purely based on neural networks could be as good as then-state-of-the-art alternatives (e.g., PBMT).



RNN Neural Machine Translation

- **Source**: *An admitting privilege is the right of a doctor to admit a patient to a hospital or a medical centre to carry out a diagnosis or a procedure, based on his status as a health care worker at a hospital.*
- **When collapsed:** *Un privilège d'admission est le droit d'un médecin de reconnaître un patient à l'hôpital ou un centre médical d'un diagnostic ou de prendre un diagnostic en fonction de son état de santé.*
- **RNNSearch**: *Un privilège d'admission est le droit d'un médecin d'admettre un patient à un hôpital ou un centre médical pour effectuer un diagnostic ou une procédure, selon son statut de travailleur des soins de santé à l'hôpital.*

RNN Neural Machine Translation

- **Source**: *An admitting privilege is the right of a doctor to admit a patient to a hospital or a medical centre to carry out a diagnosis or a procedure, based on his status as a health care worker at a hospital.*
- **When collapsed:** *Un privilège d'admission est le droit d'un médecin de reconnaître un patient à l'hôpital ou un centre médical d'un diagnostic ou de prendre un diagnostic en fonction de son état de santé.*
- **RNNSearch**: *Un privilège d'admission est le droit d'un médecin d'admettre un patient à un hôpital ou un centre médical pour effectuer un diagnostic ou une procédure, selon son statut de travailleur des soins de santé à l'hôpital.*

RNN Neural Machine Translation

- Sensible alignment between source and target tokens
- Capture long-range reordering/dependencies
- Without strong supervision on the alignment
  - Weakly supervised learning

(Bahdanau et al., 2015; Jean et al., 2015)

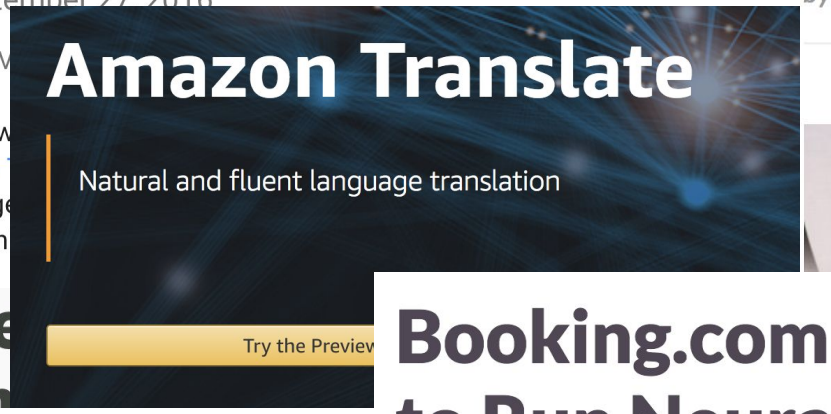
A Neural Network for Machine Translation, at Produ
Scale

Tuesday, September 27, 2016

Posted by Quoc V

Ten years ago, w
Based Machine
machine intellige
improving mach

Toda
of-th
quali



Inside the EPO's Machine-Powered Mission to Unlock Europe's Multilingual Patents

by Eden Estopace on June 6, 2017

Adoption of Neural Machine Translation (NMT) in production environments is gathering pace. In a blog post on May 15,

:PO)
:e, the
n 2013
ıte is a
ırch

Booking.com Builds on Harvard Framework to Run Neural MT at Scale

by Eden Estopace on July 31, 2017

Three major trends shaping the language technology space converged at Booking.com in what is likely a harbinger of things to come in the language industry.

Systran launch
translation eng

Language barriers represent one of th
Now, thanks to advances in artificial in

By Eileen Brown for Social Business | November

ork based

Rat

Facebook is an onlin
allows its users to co
well as make new co

Share 461

Microsoft Translator is now powering all speech translation through state-of-the-art neural networks.

In practice,

- Many excellent open-source packages exist:
  - Nematus https://github.com/EdinburghNLP/nematus
    - Compute backend: TensorFlow (originally Theano)
    - Used to build state-of-the-art translation systems for WMT'16 and WMT'17.
    - Supported by U. Edinburgh (Rico Sennrich's group)
  - OpenNMT-py https://github.com/OpenNMT/OpenNMT-py
    - Compute backend: PyTorch (originally Lua-Torch)
    - Implements latest architectures and algorithms
    - Supported by Harvard NLP (Sasha Rush's group)
  - FairSeq https://github.com/facebookresearch/fairseq
    - Compute backend: PyTorch
    - Focuses on the convolutional seq2seq
    - Supported by Facebook AI Research
  - Sockeye https://github.com/awslabs/sockeye
    - Compute backend: MXNet
    - Supported by Amazon

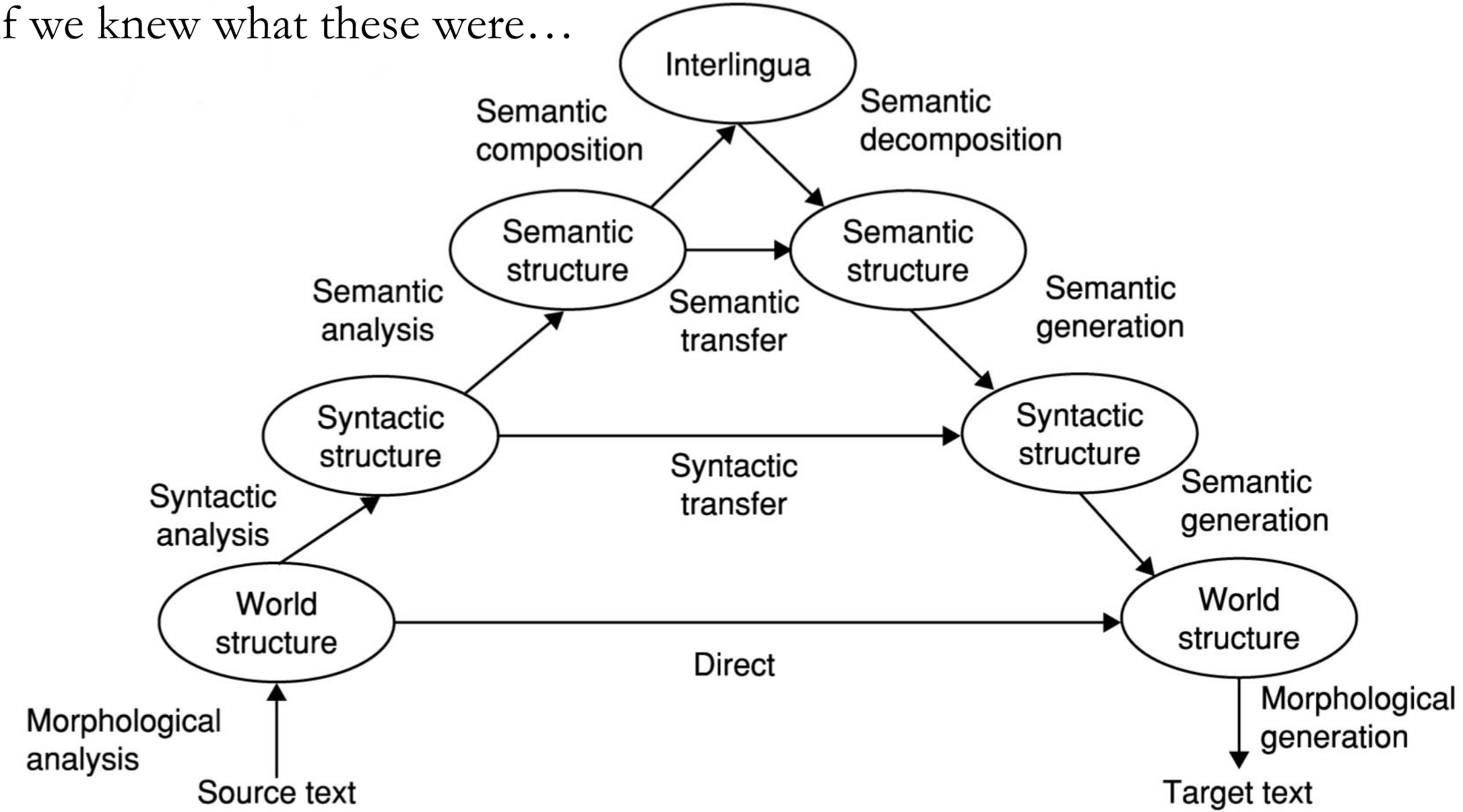
In practice,

- Many new architectures are being proposed constantly
- Convolutional sequence-to-sequence models [Gehring et al., 2017]
  - Encoder: CNN-based sentence representation
  - Decoder: CNN-based conditional language model
- Transformers [Vaswani et al., 2017]
  - Encoder: Self-attention based sentence representation
  - Decoder: Self-attention based conditional language model
- It has been four years only, and a long road lies ahead…

Intermission:
A bit of historical remark

When did neural machine translation start, and where does it fit?

Only if we knew what these were…



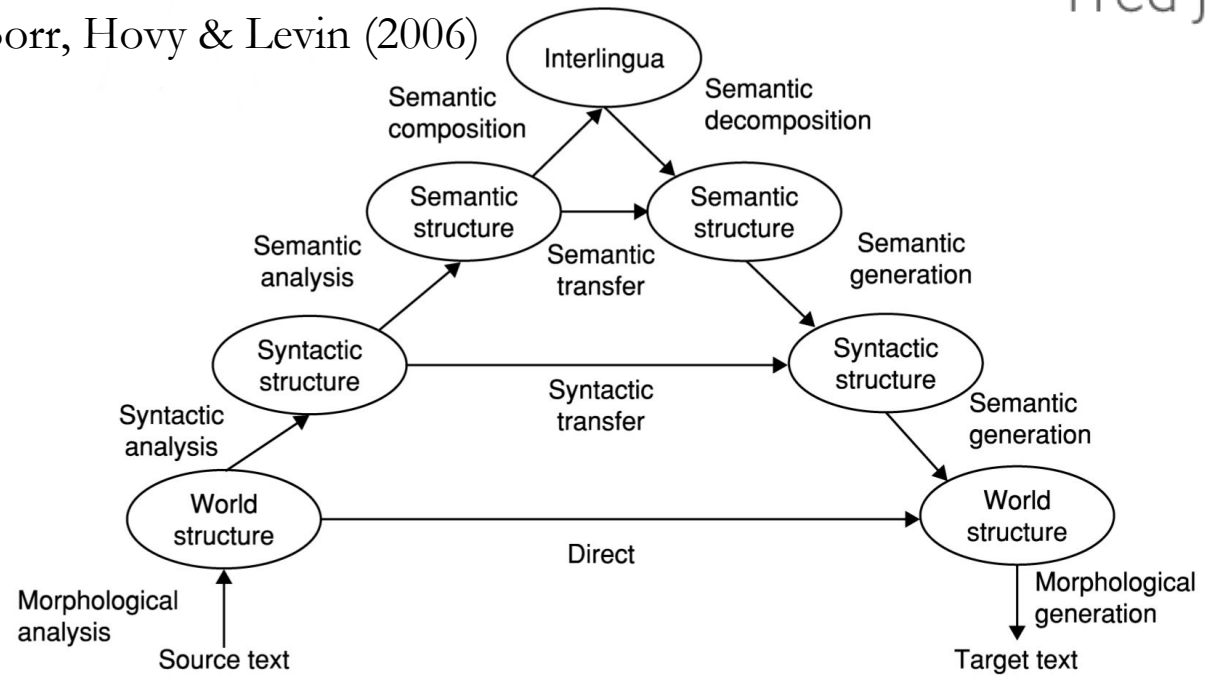
Borr, Hovy & Levin (2006)

Unfortunately not…

"Every time I fire a linguist, the performance of the recognizer goes up."

- Fred Jelinek (IBM), 1988

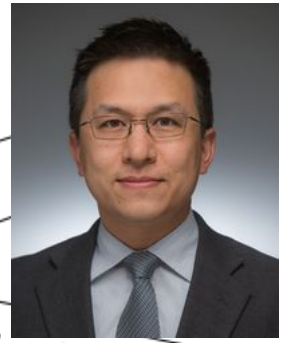
Borr, Hovy & Levin (2006)



Machine Translation

Rule-based MT

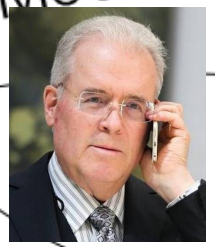
Phrase-based MT



Hierarchical Phrase MT

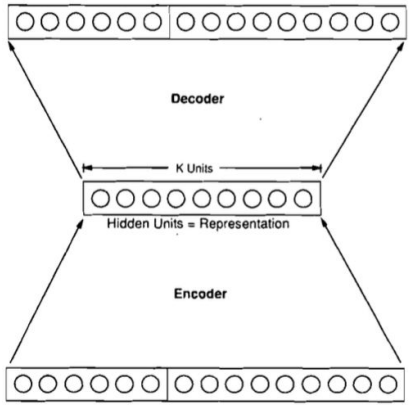
Statistical MT

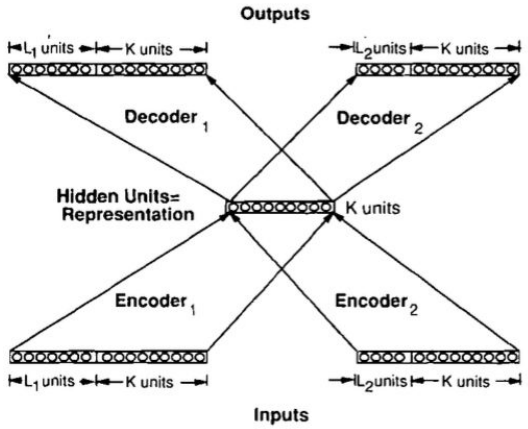
IBM Models



Syntax-based MT

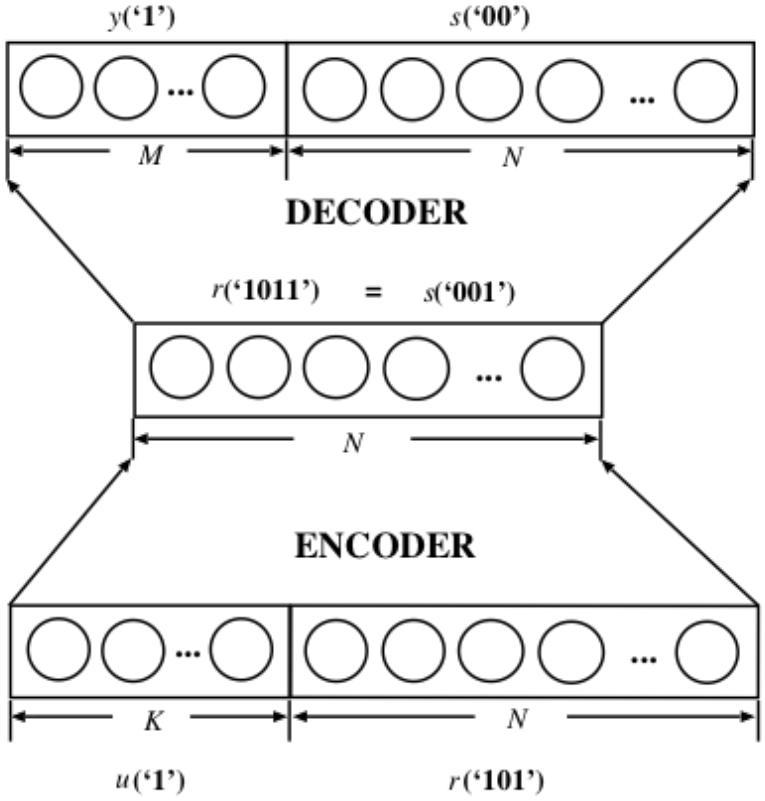
Neural MT



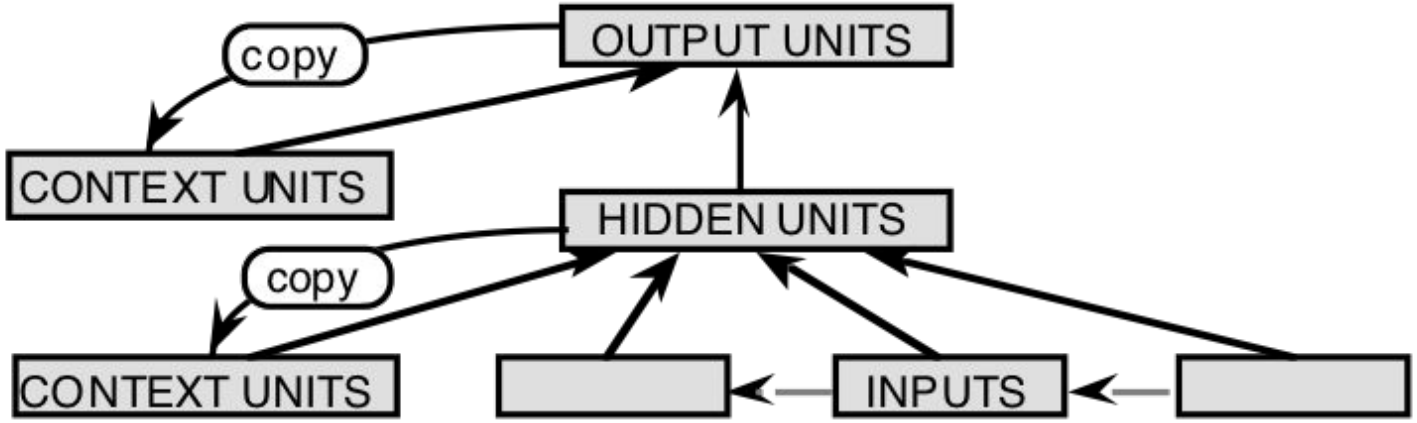


- [Allen 1987 IEEE 1st ICNN]
- 3310 En-Es pairs constructed on 31 En, 40 Es words, max 10/11 word sentence; 33 used as test set
- Binary encoding of words – 50 inputs, 66 outputs; 1 or 3 hidden 150-unit layers. Ave WER: 1.3 words

- [Chrisman 1992 *Connection Science*]
- Dual-ported RAAM architecture [Pollack 1990 *Artificial Intelligence*] applied to corpus of 216 parallel pairs of simple En-Es sentences:
- Split 50/50 as train/test, 75% of sentences correctly translated!



Brief resurrection in 1997



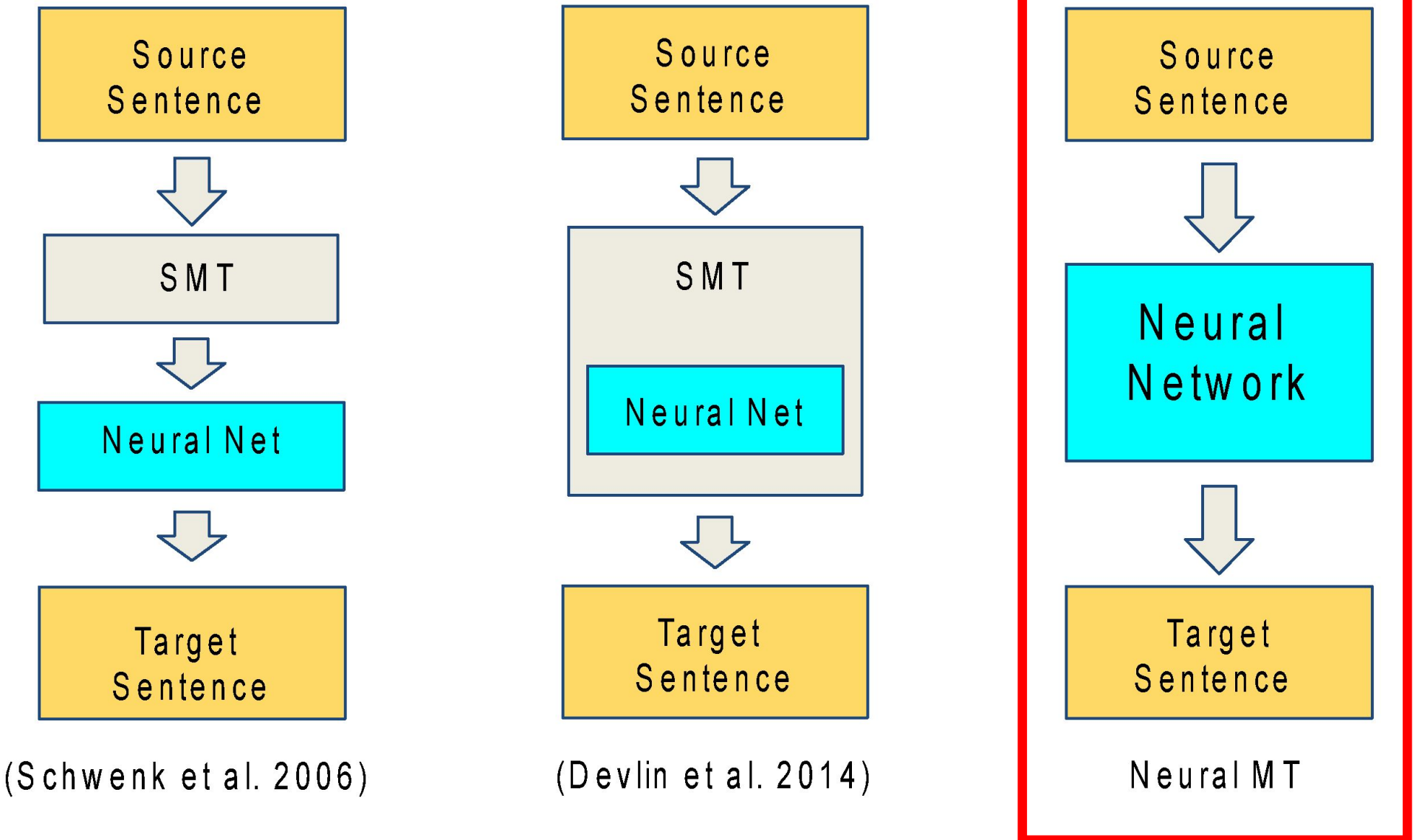
*"We propose .. **Recursive Hetero-Associative Memory** which .. may be applied **to learn general translations from examples** in which different se e the same translation."*

– Forcada & Ñeco, 1997

*"Based on these encouraging performances, future work dealing with more complex limited-domain translations seems to be feasible. **However, the size of the neural nets required for such applications (and consequently, the learning time) can be prohibitive**"*

- Castano & Casacuberta, 1997

Modern neural machine translation



When was “neural machine translation” coined?

CIFAR NCAP - Summer School 2014

Tuesday, August 12, 2014 - Saturday, August 16, 2014

Location: University of Toronto

Organizers: Geoffrey Hinton and Yoshua Bengio

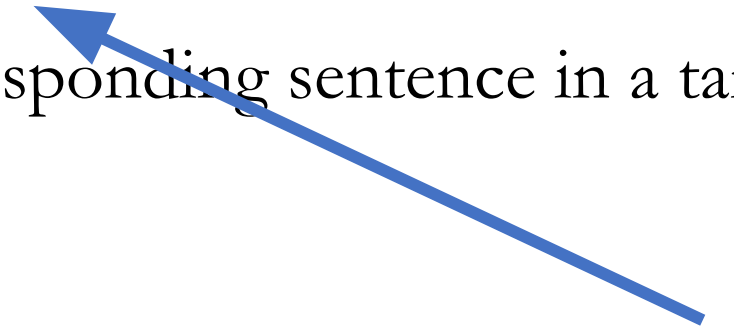
9. Many Paths to Computing Normalizing Constants by Yuri Burda (slides)
10. Impact of Attention on Perception by Ashkan Amiri (slides)
11. Neural Machine Translation: Approaches, Challenges and By-Products by Kyunghyun Cho (slides)
12. Training Neural Bayesian Nets by Laurent Dinh (slides)
13. Pixels to Voxels: Modelling Visual Representations in the Human Brain by Pulkit Agrawal (slides)
14. Error Correction as Data Modeling in Emerging Memory Technologies by Jesse Engel (slides)

What if the source is not a language?

Neural multimedia description generation – a bit old news, but worth a revisit

Machine Translation

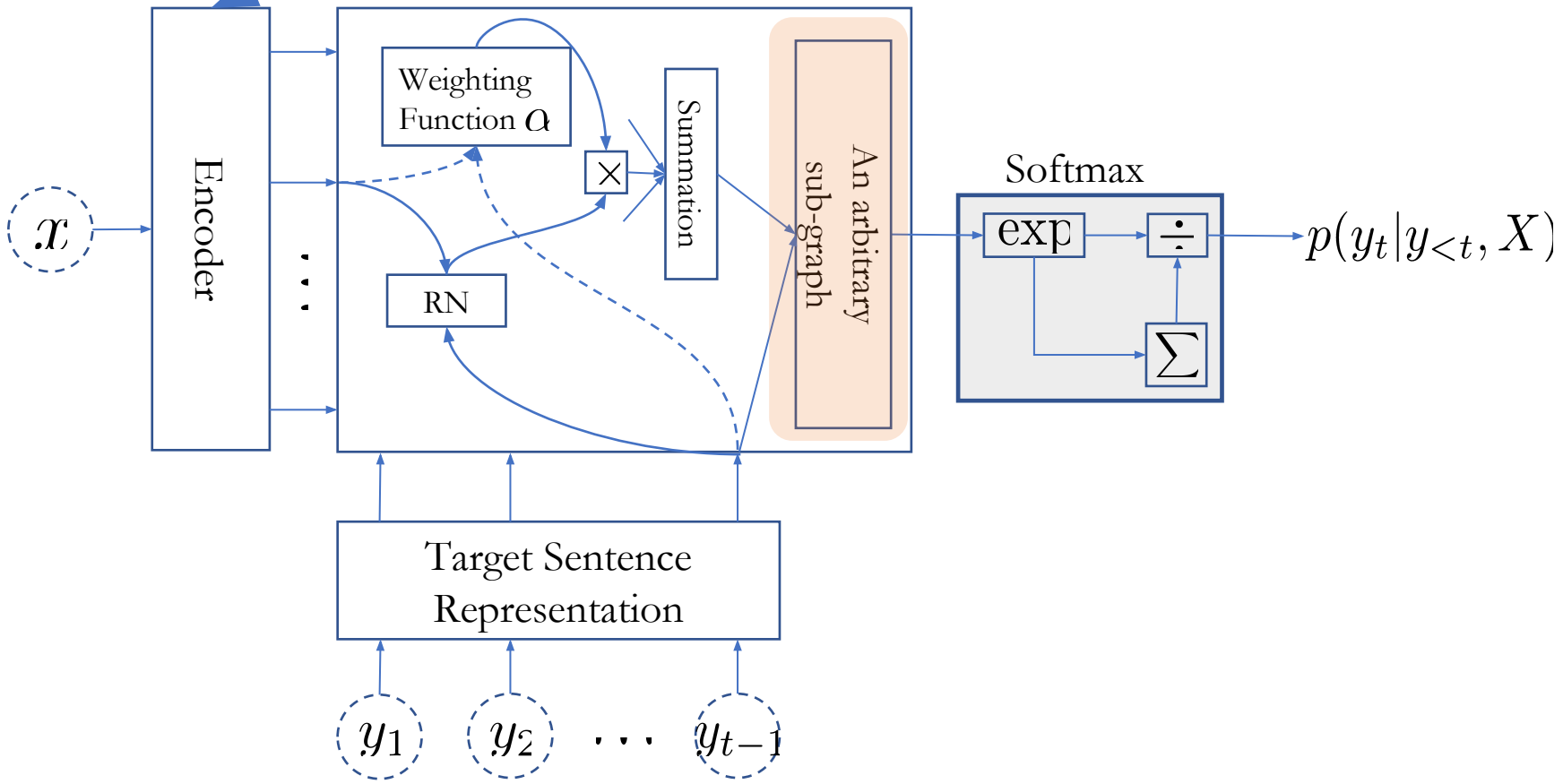
- Input: **a sentence written in a source language**
- Output: a corresponding sentence in a target language



Is it necessary for the source to be a natural language sentence?

Description Generation

- Input: **arbitrary as long as encoded into a set of continuous vectors**
- Output: a corresponding sentence in a target language



Description Generation

- Encouraged by the success of neural machine translation, a lot of new applications were tried in 2015-2016:
  - Image caption generation
  - Video description generation
  - Speech recognition
  - And many others.
- In most of these tasks, the attention-based encoder-decoder has since become *de facto* standard: see Lecture 4.

Image Caption Generation [Xu et al., 2015

- Input: an image
- Output: an image caption
- Network Architecture
  - Encoder: deep convolution network
  - Decoder: recurrent language model with the attention mechanism.
- Data: image-caption pairs

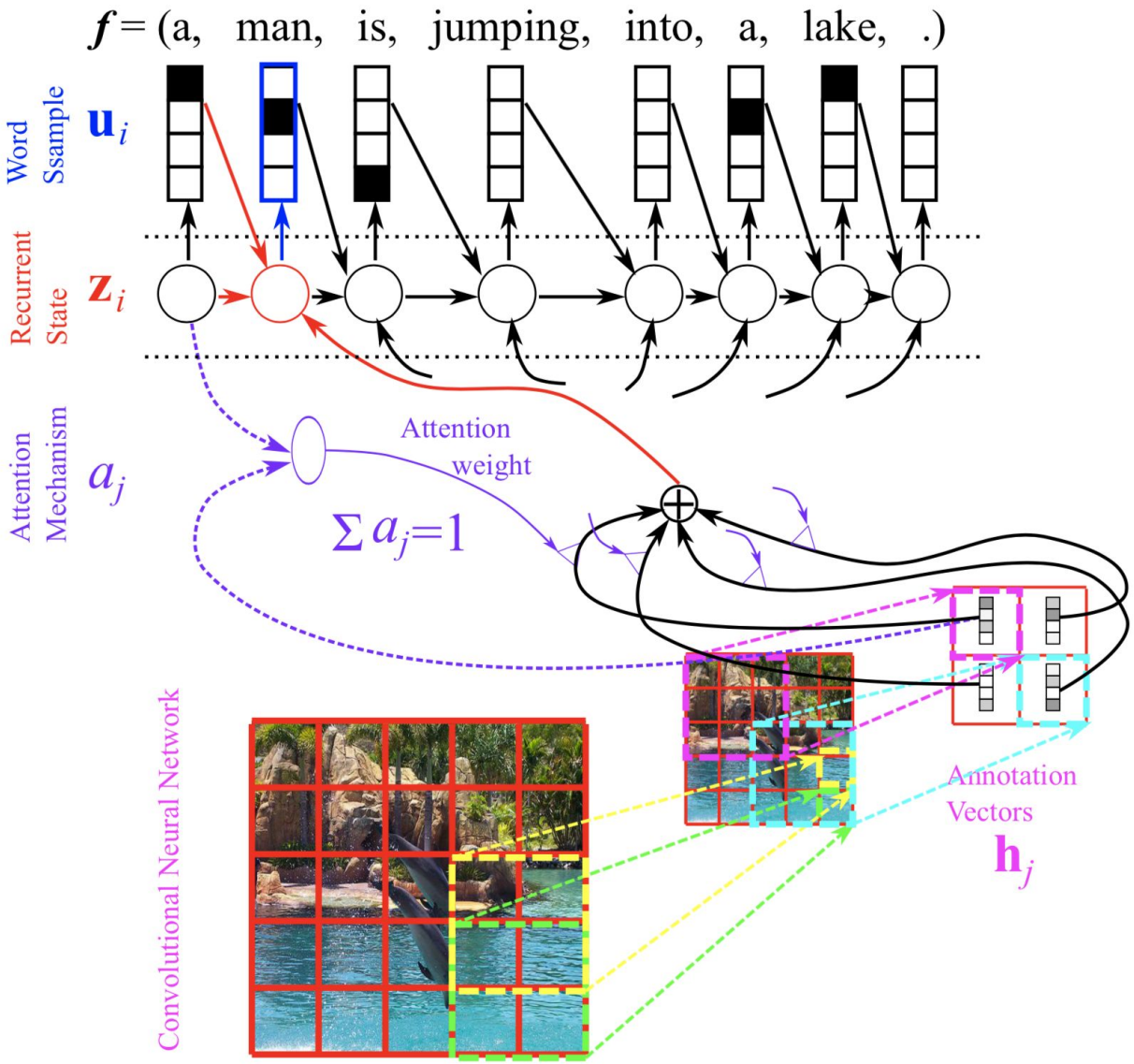


Image Caption Generation

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

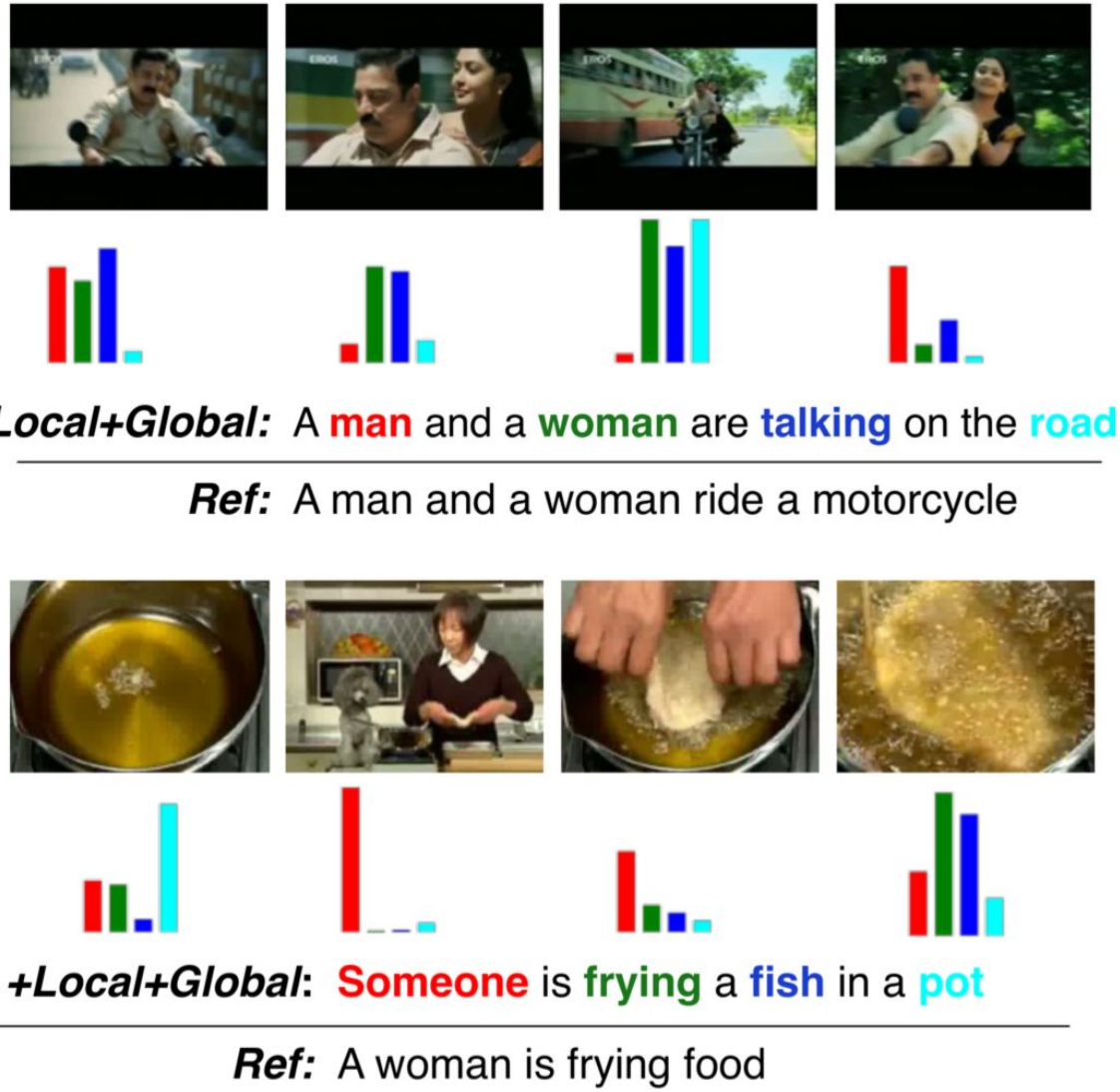
A giraffe standing in a forest with trees in the background.

Video Description Generation [Li et al., 2015]

- Input: a short video clip - a sequence of video frames.
- Output: a corresponding description
- Network Architecture
  - Encoder: a deep 2+3D convolutional network
    1. A 2-D convolutional network for each frame
    2. A 3-D convolutional network for the entire clip
  - Decoder: recurrent language modelwith the attention mechanism.
- Data: clip-description pairs – collected from YouTube.

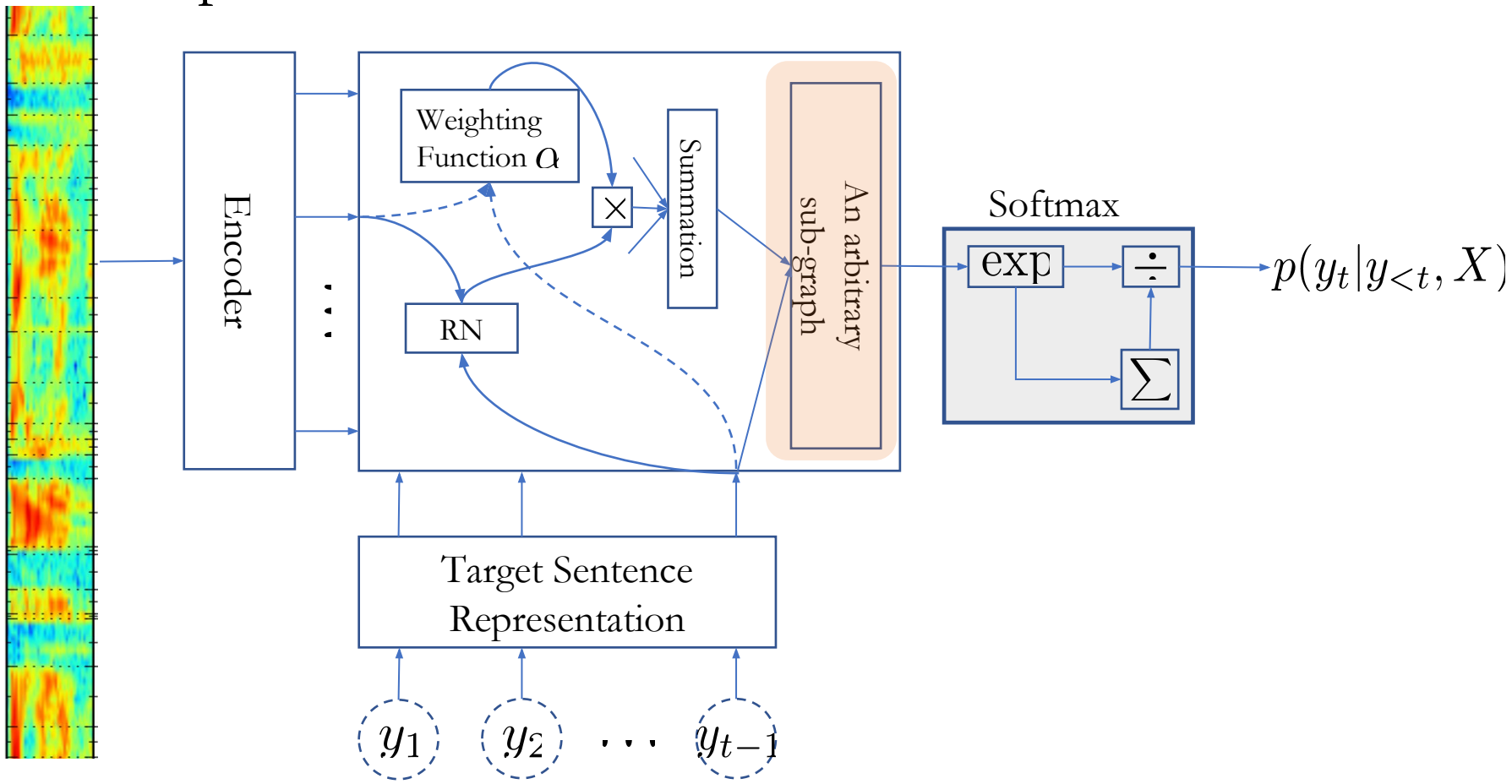
Video Description Generation

- Input: a short video clip - a sequence of video frames.
- Output: a corresponding description
- Attention allows us to inspect the inner-working of the model.
- Some encouraging result in 2015, and a lot of advances have been proposed since then.



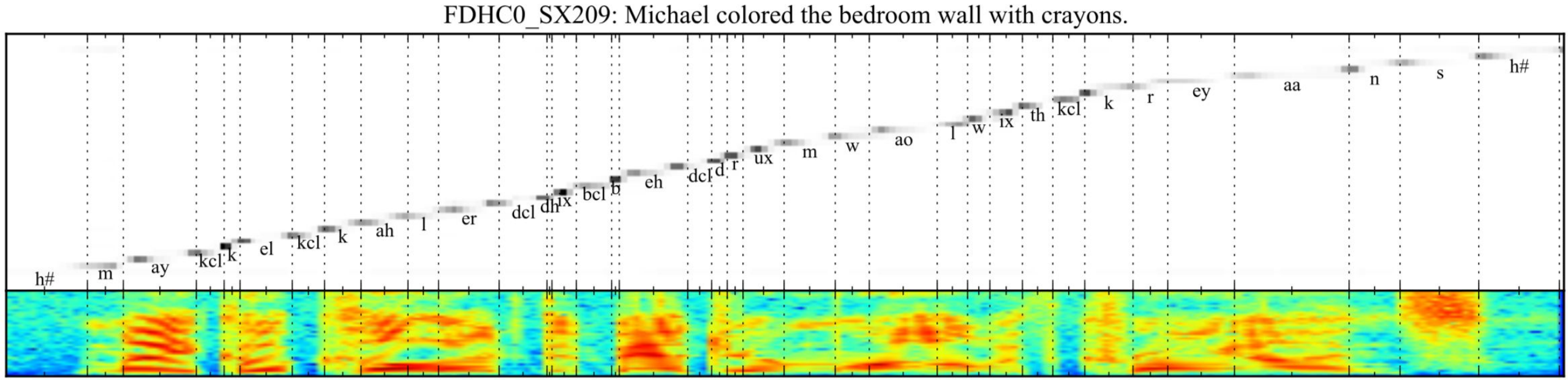
Speech Recognition [Chorowski et al., 2015]

- Input: Speech
- Output: transcription



Speech Recognition

- Input: Speech
- Output: transcription
- Network Architecture
  - Encoder: convolution+recurrent acoustic network
  - Decoder: conditional recurrent language model + attention mechanism



Since 2015…

- The attention (alignment) mechanism has become a work horse behind various AI models/applications including
  - Neural Turing machines (differentiable neural computer) [Graves et al., 2015&2016], memory networks [Weston et al., 2015; Sukhbaatar et al., 2016], dynamic neural Turing machines [Gulcehre et al., 2017; Miller et al., 2017], …
  - Reinforcement learning: attentive history selection [Tian et al., 2016], neural episodic control [Pritzel et al., 2017]
  - Generative models: DRAW [Gregor et al., 2016], Image Transformer [Parmar et al., 2018], …
- Now *de facto* **standard**

What if there are many languages?

Multilingual machine translation and meta-learning

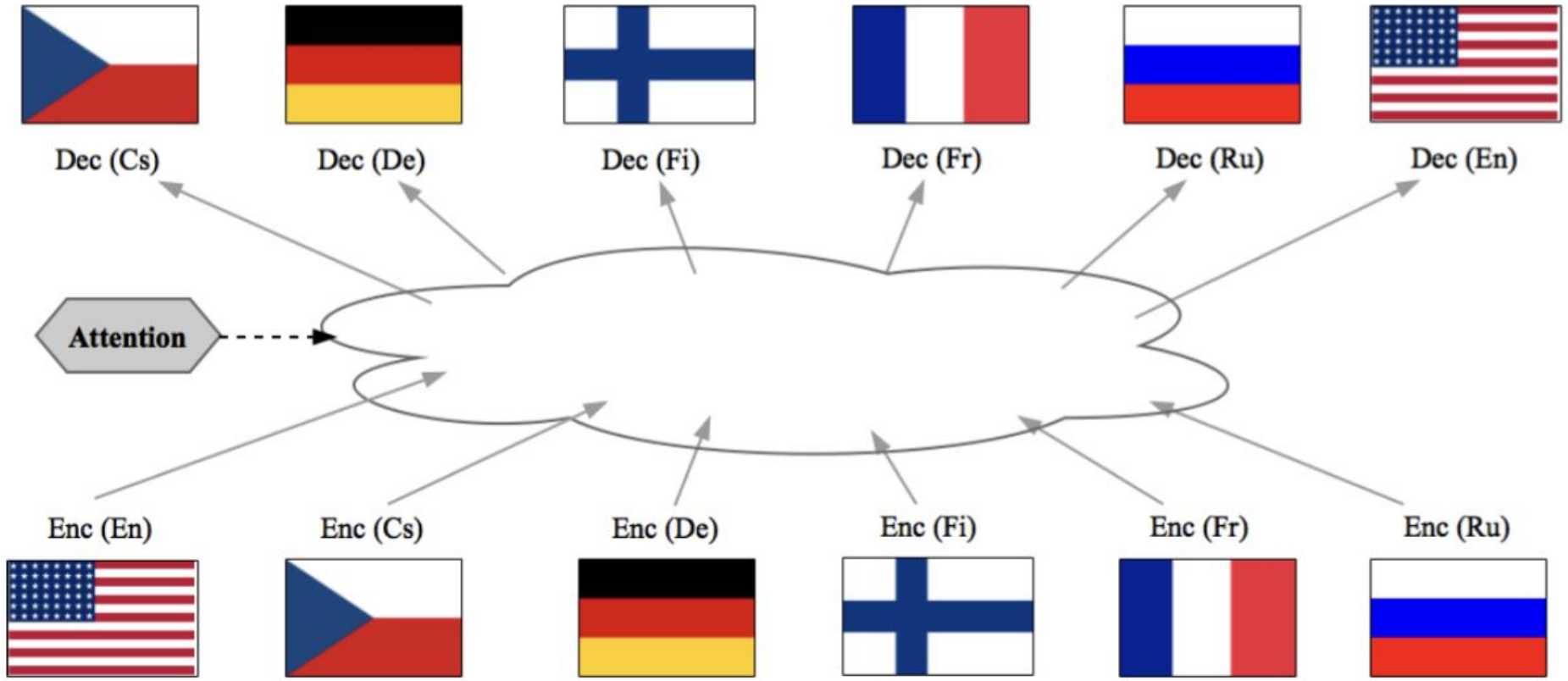
Multilingual Translation and Meta-Learning

Multilingual Translation – (1)

- Traditionally,
  - If a parallel corpus exists, one system for each language pair.
    - Parallel corpus: $D^{a \to b} = \left\{ (X_1^a, Y_1^b), \ldots, (X_N^a, Y_N^b) \right\}$
    - Translation system: $\log p(Y^b | X^a)$
  - If no direct parallel corpus exists, a pivot-based translation.
    - No direct parallel corpus: $D^{a \to b} = \emptyset$
    - But, $|D^{a \to c}| > 0, |D^{c \to b}| > 0$
    - Then, $\log p(Y^b | \hat{X}^c)$, where $\hat{X}^c = \arg\max_X \log p(X^c | X^a)$
    - *c* is a pivot language (often, English.)
  - No knowledge transfer between different language pairs.

Multilingual Translation as Multitask Learning – (2)

- Now, [Firat et al., 2016a; Firat et al., 2016b; Johnson et al., 2016; Ha et al., 2016; Lee et al., 2017]



Multilingual Translation as Multitask Learning – (3)

- Separate encoder/decoders
  - [Firat et al., 2016a; Firat et al., 2016b]
- One encoder per source *l*

$$f^l_{\text{enc}} : V_l \times \cdots \times V_l \to \mathbb{R}^d \times \cdots \times \mathbb{R}^d$$

- One decoder per target *l'*

$$\log p^{l'}(Y^{l'}|H)$$

- For each pair (l, l'),

$$\log p^{l'}(Y^{l'}|H = f^l_{\text{enc}}(X^l))$$

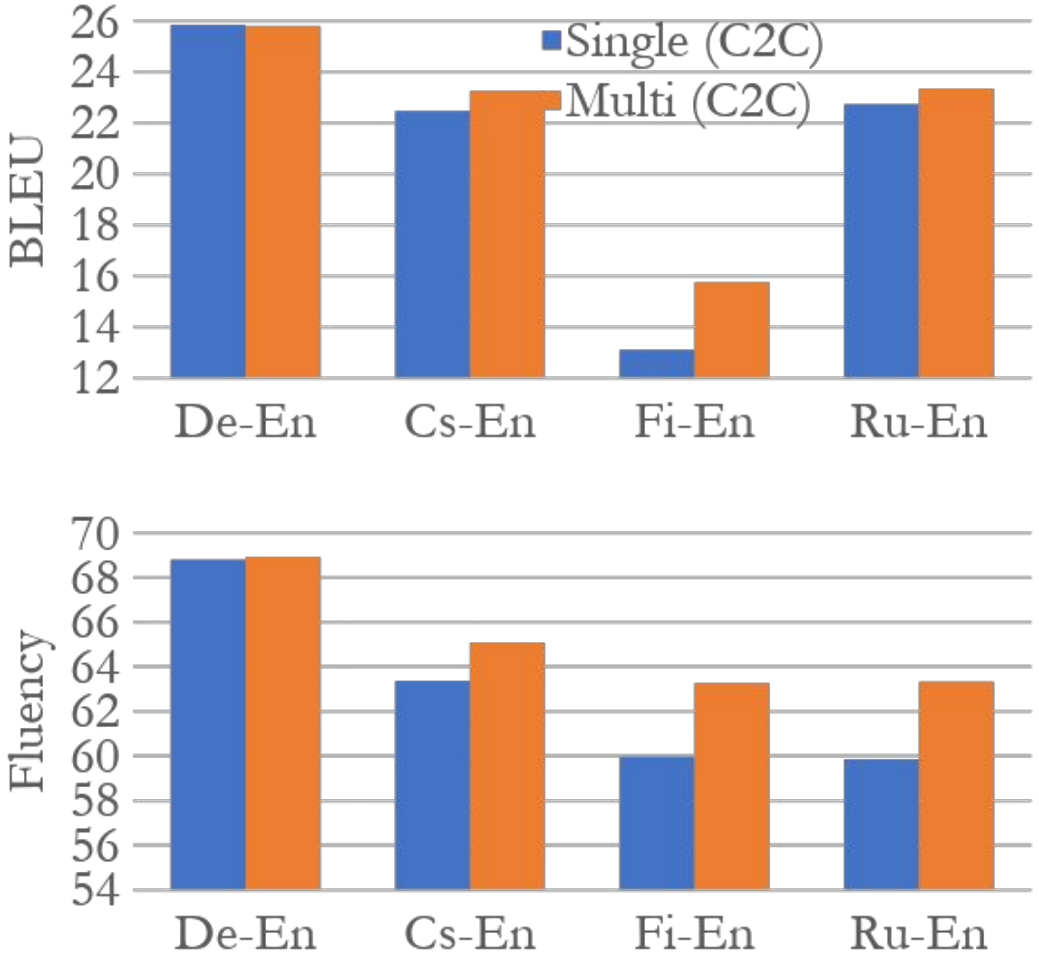
- Train using all available language pairs

- Universal encoder/decoders
  - [Johnson et al., 2016; Ha et al., 2016; Lee et al., 2017; Gu et al., 2018]
  - With hypernetworks [Platanios et al., 2018]
- Shared lexicons $f^l_{\text{lex}} : V_l \to V$
  - A shared vocabulary of language-agnostic tokens [J, 2016; H, 2016; L, 2017]
  - Universal lexical representation [G, 2018]
- One encoder-decoder for all pairs

$$f^{-l'}_{\text{lex}}(\arg\max_Y \log p(Y|f^l_{\text{lex}}(X^l)))$$

Multilingual Translation as Multitask Learning – (4)

- Does it work?
- Single-pair Systems
  De→En, Cs→En, Fi→En, Ru→En
- Multilingual System
  {De, Cs, Fi, Ru}→En
- The latter has 1/4x parameters
- Better translation quality on
  low-resource languages (Fi & Ru)



Lee, Cho and Hoffman (2017)

Multilingual Translation as Multitask Learning – (5)

- Does it work? – **Yes!***
- Single-pair Systems vs. Multilingual System
- Works with intra-sentence code-switching

(e) Multilingual

Multi src	Bei der Metropolitního výboru pro dopravu für das Gebiet der San Francisco Bay erklärten Beamte , der Kongress könne das Problem банкротство доверительного Фонда строительства шоссейных дорог einfach durch Erhöhung der Kraftstoffsteuer lösen .
EN ref	At the Metropolitan Transportation Commission in the San Francisco Bay Area , officials say Congress could very simply deal with the bankrupt Highway Trust Fund by raising gas taxes .
bpe2char	During the Metropolitan Committee on Transport for San Francisco Bay , officials declared that Congress could solve the problem of bankruptcy by increasing the fuel tax bankrupt .
char2char	At the Metropolitan Committee on Transport for the territory of San Francisco Bay , officials explained that the Congress could simply solve the problem of the bankruptcy of the Road Construction Fund by increasing the fuel tax .

** It often fails to translate between a pair of languages not seen during training*

Lee, Cho and Hoffman (2017)

Limitations of Multitask Learning – (1)

- Tricky when the availability of data drastically differs across languages.
  - *overfitting* on low-resource pairs, while *underfitting* on high-resource pairs.

$$L(\theta) = \sum_{l} \frac{1}{N^l} \sum_{n=1}^{N^l} \log p_\theta(Y_n^l | X_n^l)$$

  - Extremely low-resource pairs can easily be *ignored*.

$$L(\theta) = \sum_{l} \sum_{n=1}^{N^l} \log p_\theta(Y_n^l | X_n^l)$$

  - See [Firat et al., 2016a] and [Lee et al., 2017] for more discussion.
  - *It is really horrible to figure out how to tackle this in practice…*

Limitations of Multitask Learning – (2)

- Assumes the availability of all language pairs in advance.
  - The entire model must be re-trained each time a new language is introduced.
- Transfer Learning [Zoph et al., 2016; Nguyen & Chiang, 2017]
  - Only re-train a subset of parameters on a new language pair.
  - Many possible strategies, but no clear winning strategy.

Setting	Dev BLEU	Dev PPL
No retraining	0.0	112.6
Retrain source embeddings	7.7	24.7
+ source RNN	11.8	17.0
+ target RNN	14.2	14.5
+ target attention	**15.0**	13.9
+ target input embeddings	14.7	**13.8**
+ target output embeddings	13.7	14.4

Zoph et al., 2016

Limitation of Multitask Learning – (3)

- Inconvenient truths about multitask+transfer learning
  - Relies on our intuition that all languages/tasks share common underlying structures: *true?*
  - Assumes multitask learning can capture those underlying structures and share across multiple languages/tasks: *true?*
  - Assumes multitask-learned parameters are a good initialization for further training: *true?*

- Is there a more satisfying approach?

Meta-Learning: MAML [Finn et al., 2018] – (1)

- Model-agnostic meta-learning [Finn et al., 2018]
- Two-stage learning
  1. Simulated learning

$$\begin{aligned}\mathrm{Learn}(D_{\mathcal{T}};\theta^0) &= \arg\max_{\theta} \mathcal{L}^{D_{\mathcal{T}}}(\theta) \\ &= \arg\max_{\theta} \sum_{(X,Y)\in D_{\mathcal{T}}} \log p(Y|X,\theta) - \beta\|\theta-\theta^0\|^2,\end{aligned}$$

  2. Meta-learning

$$\mathcal{L}(\theta) = \mathbb{E}_k \mathbb{E}_{D_{\mathcal{T}^k}, D'_{\mathcal{T}^k}} \left[\sum_{(X,Y)\in D'_{\mathcal{T}^k}} \log p(Y|X;\mathrm{Learn}(D_{\mathcal{T}^k};\theta)) \right],$$

Meta-Learning: MAML [Finn et al., 2018] – (2)

1. Simulated learning
   - Given a small subset $D_{\mathcal{T}}$ of the training set of task $\mathcal{T}$, update the model parameters $N = 1$ times.

$$\begin{aligned}\text{Learn}(D_{\mathcal{T}}; \theta^0) &= \arg\max_{\theta} \mathcal{L}^{D_{\mathcal{T}}}(\theta) \\ &= \arg\max_{\theta} \sum_{(X,Y)\in D_{\mathcal{T}}} \log p(Y|X,\theta) - \beta\|\theta - \theta^0\|^2, \\ &= \theta_0 + \eta\nabla_{\theta}\mathcal{L}^{D_{\mathcal{T}^k}}(\theta_0)\end{aligned}$$

   - Clip the update so that $\eta\nabla_{\theta}\mathcal{L}^{D_{\mathcal{T}^k}}(\theta_0)$ does not deviate too much from θ_C.
   - *It simulates finetuning on a target task with a limited resource.*

Meta-Learning: MAML [Finn et al., 2018] – (3)

2. Meta-Learning
   - Randomly select a task k and select a training subset $D = D_{\mathcal{T}^k}$.
   - Randomly select a validation subset $D' = D'_{\mathcal{T}^k}$ for evaluation.
   - Update the meta-parameter θ_0 by gradient descent:

$$\theta_0 \leftarrow \theta_0 + \eta_0 \nabla_{\theta_0} \mathcal{L}^{D'}(\theta_0),$$

   where

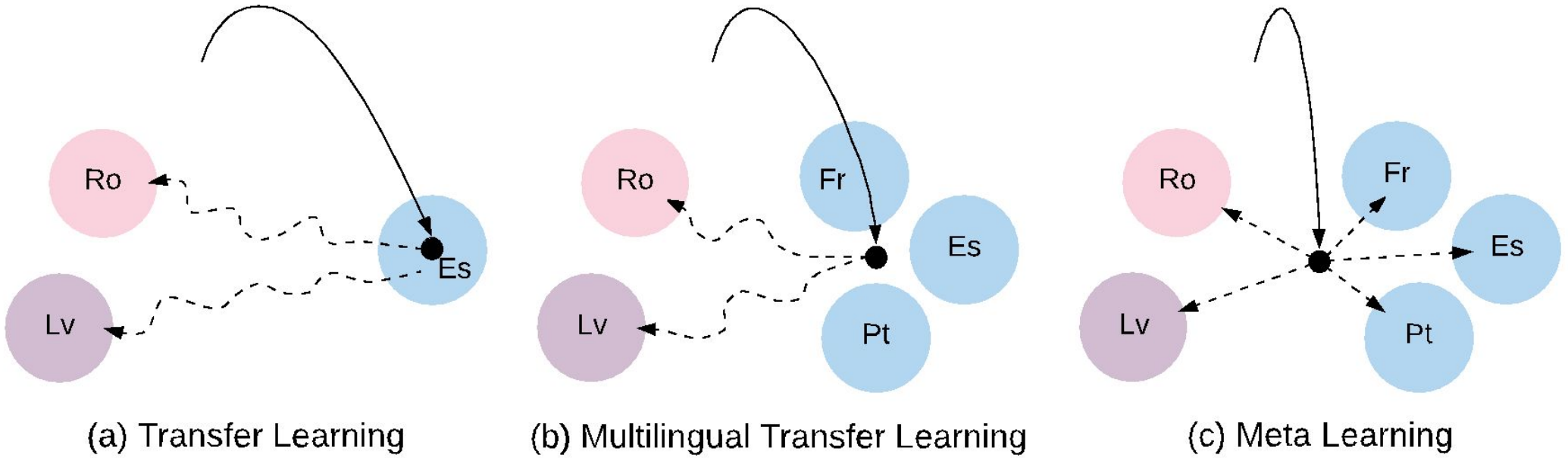
$$\begin{aligned}\nabla_\theta \mathcal{L}^{D'}(\theta') &= \nabla_{\theta'} \mathcal{L}^{D'}(\theta') \nabla_\theta (\theta - \eta \nabla_\theta \mathcal{L}^{D}(\theta)) \\ &= \nabla_{\theta'} \mathcal{L}^{D'}(\theta') - \eta \nabla_{\theta'} \mathcal{L}^{D'}(\theta') H_\theta(\mathcal{L}^{D}(\theta))\end{aligned}$$

   - *Update the meta-parameter so that N-step GD on the k-th task works well.*

Meta-Learning: MAML [Finn et al., 2018] – (4)

3. Fast adaptation to a new task
   - Given a small training set D of the new target task, SGD starting from the meta-parameter θ_C.
   - Early stopping based on $\|\theta - \theta_0\|^2$.

Multitask learning vs. Meta-learning



a) Transfer learning does not take into account subsequent learning.

b) Multilingual learning does not take into account new, future tasks.

c) Meta-learning considers subsequent learning on new, future tasks.

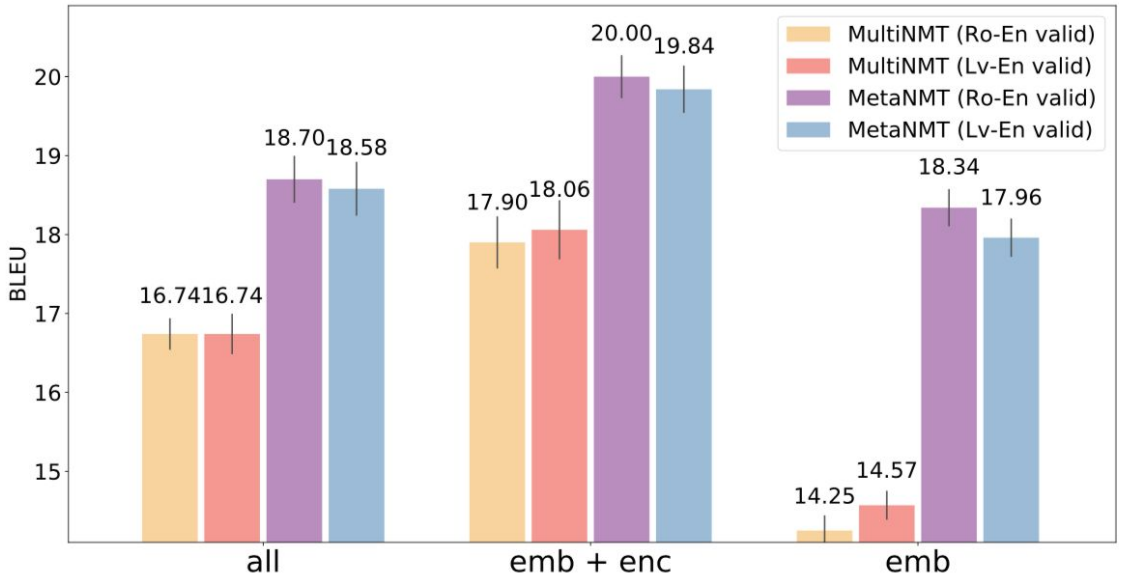
Extension to Neural Machine Translation

- I/O mismatch between different tasks
  - Vocabulary mismatch among different languages
- Multilingual word embedding [Artetxe et al., 2017; Conneau et al., 2018; and more]
  - Project each token into a continuous vector space $f^l : V^l \rightarrow \mathbb{R}^d$
  - Ensure that they are compatible:
    $\|f^l(v^l) - f^{l'}(v^{l'})\|^2 < \epsilon$, iff v^l and $v^{l'}$ have the same meaning.
- Universal lexical representation [Gu et al., 2018]

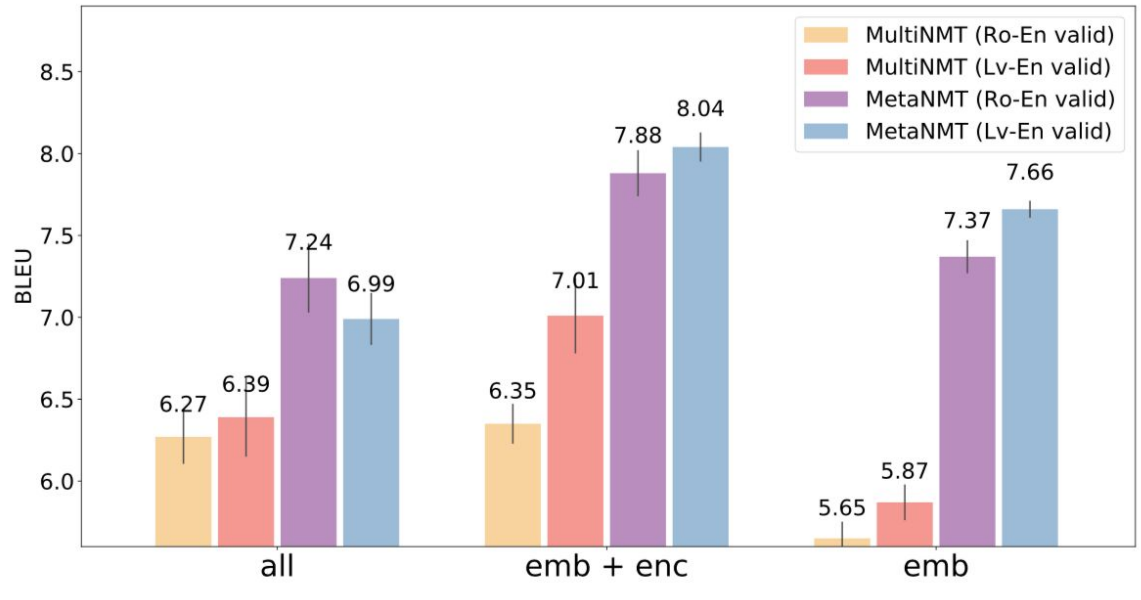
- *Meta-NMT!*

Experiments

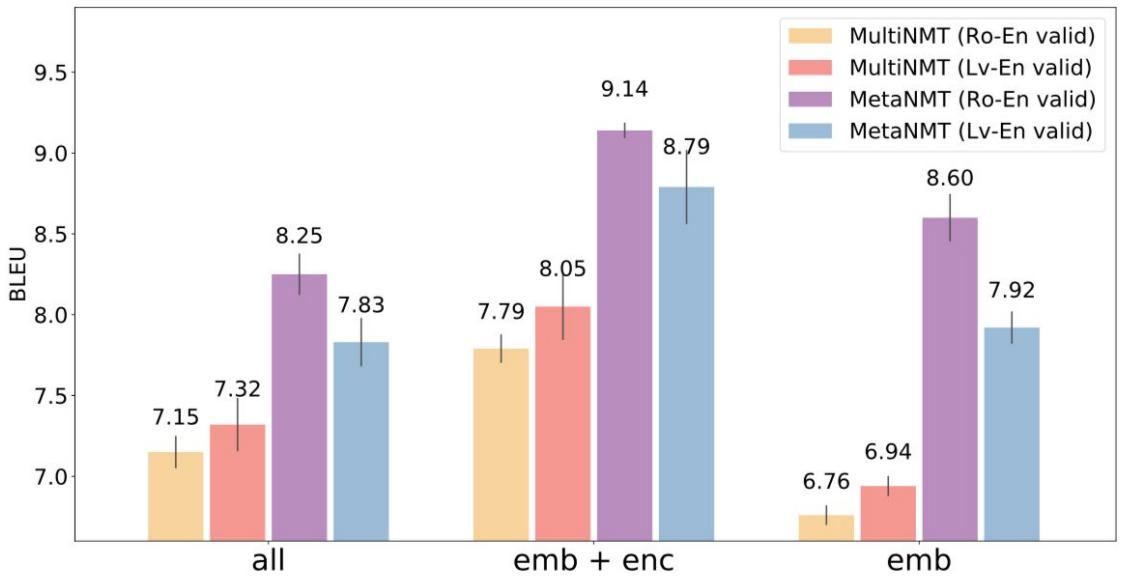
- Source tasks: all the languages from Europarl + Russian
  - Bg→En, Cs→En, Da→En, De→En, El→En, Es→En, Et→En, Fr→En, Hu→En, It→En, Lt→En, Nl→En, Pl→En, Pt→En, Sk→En, Sl→En, Sv→En and Ru→En.
  - Reasonable high-resource language pairs.
- Target tasks: (simulated) low-resource language pairs
  - Ro→En, Lv→En, Fi→En, Tr→En and Ko→En
  - Approximately 16k target tokens (English side): roughly 800 sentence pairs.
- Universal lexical representation: obtained from Wikipedia.
- Early stopping of meta-learning: either Ro-En or Lv-En



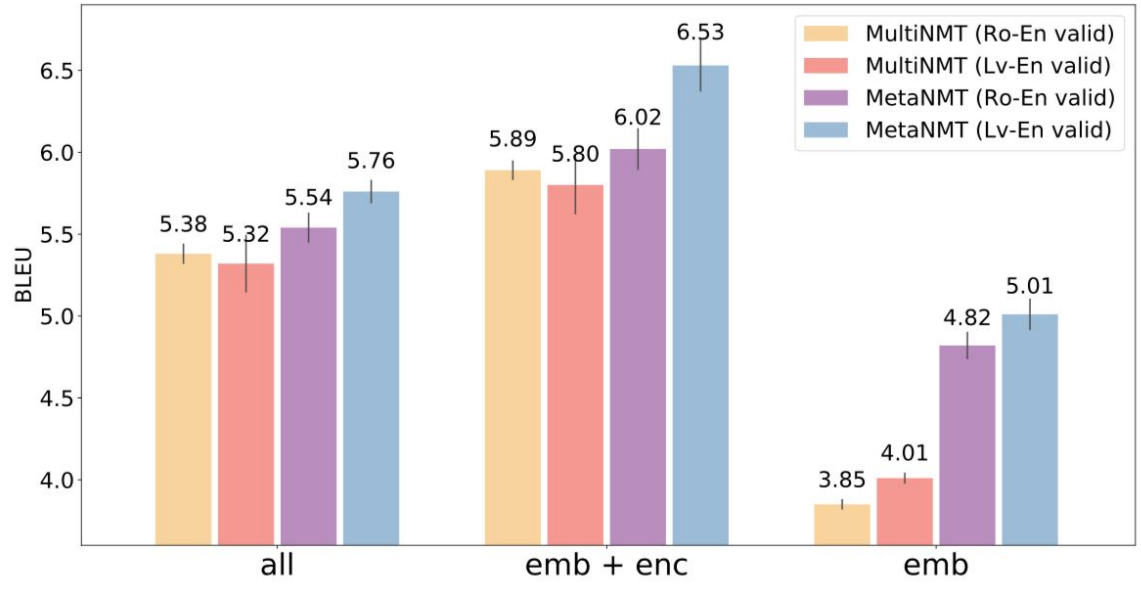
(a) Ro-En



(b) Lv-En



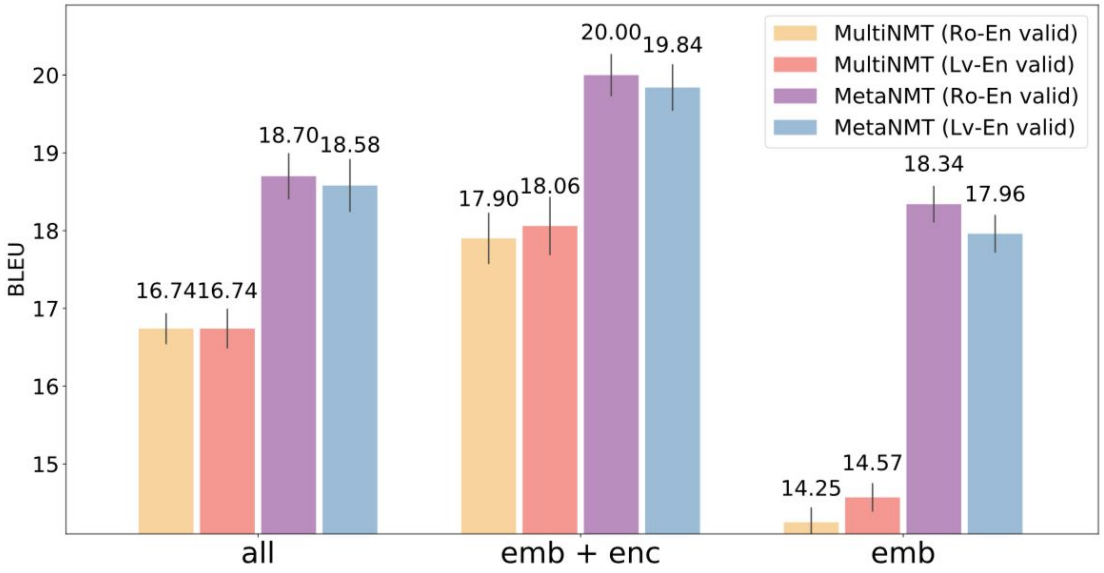
(c) Fi-En



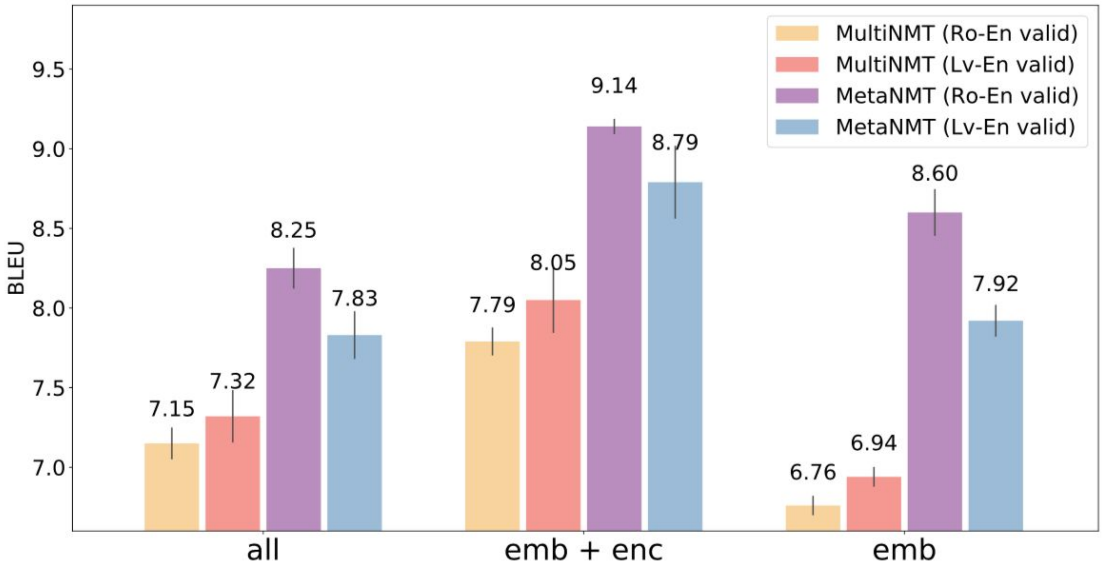
(d) Tr-En

Experiments – (1)

- Meta-learning outperforms multitask learning across all the target languages and across different finetuning strategies.
- Using only 800 examples, reaches up to 65% of fully-supervised models in terms of BLEU.



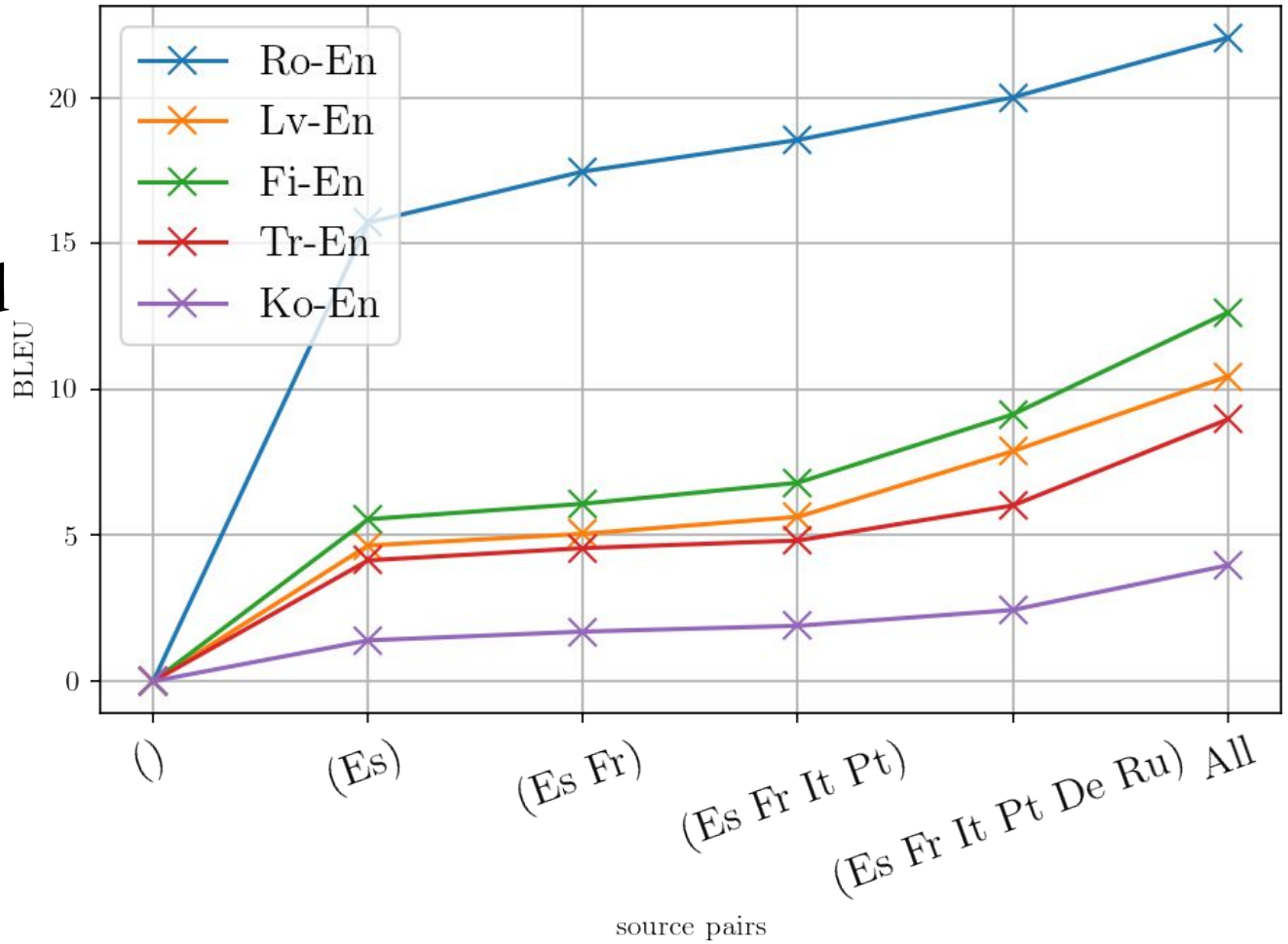
(a) Ro-En



(c) Fi-En

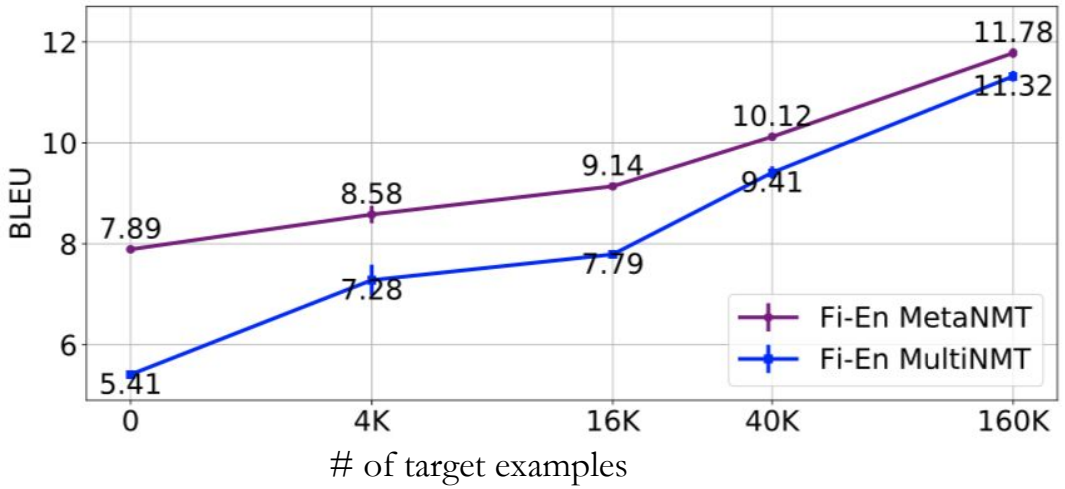
Experiments – (2)

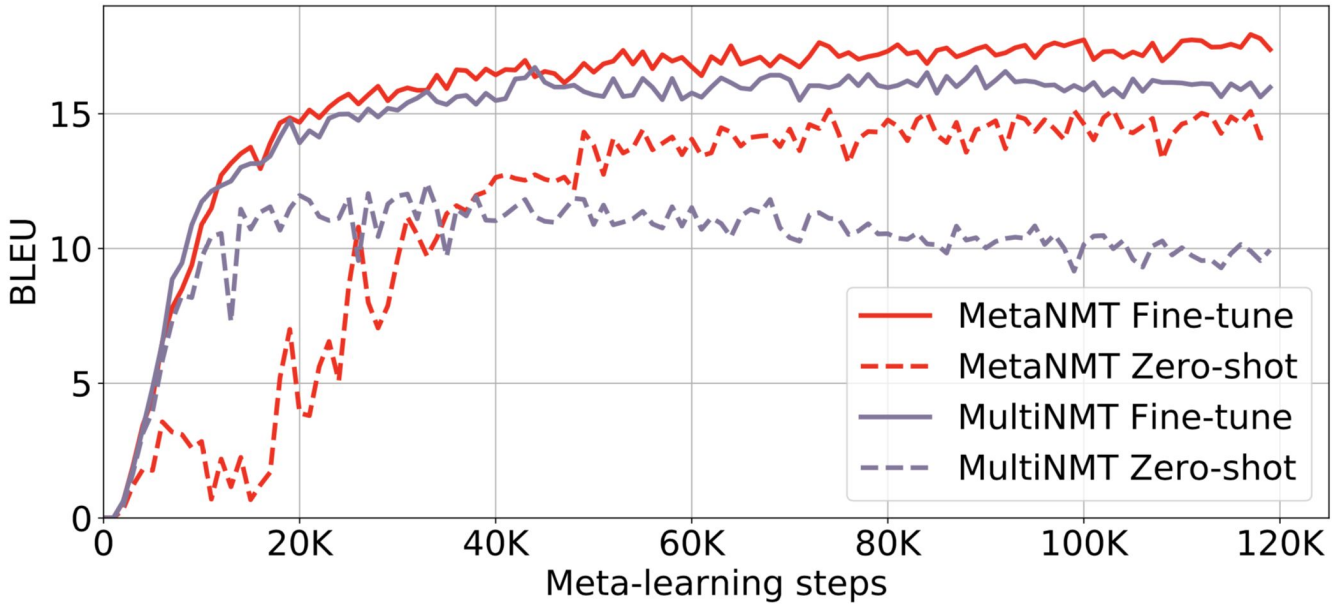
- More source tasks lead to greater improvements.
- The similarity between source and target asks matters.



Experiments – (3)

- Multi-task learning over-adapts to the source tasks.
  - Performance on the target task degrades with longer multi-task learning.
- Meta-learning does not over-adapt.
  - The meta-learning objective explicitly takes into account finetuning on a target task.
  - It requires less target examples.





Experiments – (4) Sample Translations

Source (Tr)	google mülteciler için 11 milyon dolar toplamak üzere bağış eşleştirme kampanyasını başlattı .
Target	google launches donation-matching campaign to raise $ 11 million for refugees .
Meta-0	google refugee fund for usd 11 million has launched a campaign for donation .
Meta-16k	google has launched a campaign to collect $ 11 million for refugees .
Source (Ko)	이번에 체포되어 기소된 사람들 중에는 퇴역한 군 고위관리 , 언론인 , 정치인 , 경제인 등이 포함됐다
Target	among the suspects are retired military officials , journalists , politicians , businessmen and others .
Meta-0	last year , convicted people , among other people , of a high-ranking army of journalists in economic and economic policies , were included .
Meta-16k	the arrested persons were included in the charge , including the military officials , journalists , politicians and economists .

Today we have covered

- Recurrent language modeling
- Neural machine translation
- Neural multimedia description generation
- Multilingual machine translation (and meta-learning)

- *All possible because of continuous vector representation!*